(4

THEORETICAL STUDIES OF A TRANSIENT STIMULATED RAMAN AMPLIFIER

Contract N00014-86-C-2341

SAIC Report No. 88/1674

by

Curtis R. Menyuk and Godehard Hilfer

April 19, 1988

Science Applications International Corporation

4

THEORETICAL STUDIES OF A TRANSIENT STIMULATED RAMAN AMPLIFIER

Contract N00014-86-C-2341
SAIC Report No. 88/1674

by

Curtis R. Menyuk and Godehard Hilfer

April 19, 1988

SAIC
Science Applications International Corporation

Post Office Box 1303, 1710 Goodridge Drive, McLean, Virginia 22102, (703) 821-4300

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS N/A
2a. SECURITY CLASSIFICATION AUTHORITY N/A 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT Unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SAIC Report #88/1674	5. MONITORING ORGANIZATION REPORT NUMBER(S) N00014-86-C-2341
6a. NAME OF PERFORMING ORGANIZATION Science Applications Int'l Corporation 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION DCASMA - San Diego
6c. ADDRESS (City, State, and ZIP Code) 1710 Goodridge Drive McLean, VA 22102	7b. ADDRESS (City, State, and ZIP Code) 7675 Dagget Street, Suite 200/300 San Diego, CA 92111-2241
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Naval Research Laboratory 8b. OFFICE SYMBOL (If applicable) N/A	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-C-2341
8c. ADDRESS (City, State, and ZIP Code) Scientific Office 4555 Overlook Avenue, S.W. Washington, D.C. 20375-5000	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Theoretical Studies of a Transient Stimulated Raman Amplifier

12. PERSONAL AUTHOR(S)
Menyuk, Curtis Robert and Hilfer, Godehard

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 11/86 TO 04/88	1988 April 19	339

16. SUPPLEMENTARY NOTATION
N/A

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Stimulated Raman, Nonlinear Optics, Light Amplification, Light Propagation

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This final report summarizes Science Applications International Corporation's performance on contract no. N00014-86-C-2341 for the Naval Research Laboratory. Our principle deliverable, the codes RAM2D1 and PRAM1 have been completed on schedule and run successfully. Their operation is described in detail in this report as well as their application to cases of experimental importance. We have also carried out a number of analytical calculations in order to obtain greater insight into the code operation and the experiments and to make predictions in regimes of possible experimental interest which have not yet been explored. Some of these calculations were carried out in collaboration with Dr. John Reintjes of the Naval Research Laboratory. These calculations are summarized in this report. Relevant publications and presentations are also included.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Curtis R. Menyuk	22b. TELEPHONE (Include Area Code) (703) 734-5957	22c. OFFICE SYMBOL N/A

DD FORM 1473, 84 MAR
83 APR edition may be used until exhausted.
All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

FINAL REPORT

SAIC Report No. 88/1674

THEORETICAL STUDIES OF A TRANSIENT STIMULATED RAMAN AMPLIFIER

April 19, 1988

by:

Curtis R. Menyuk and Godehard Hilfer

Applied Physics Operation
Science Applications International Corporation
1710 Goodridge Drive
McLean, VA 22102

Prepared under:

Contract N00014-86-C-2341

For:

Dr. John Reintjes (Code 6542)

Laser Physics Branch
Naval Research Laboratory
4555 Overlook Avenue, SW
Washington, DC 20375-5000

THEORETICAL STUDIES OF A TRANSIENT STIMULATED RAMAN AMPLIFIER

TABLE OF CONTENTS

ABSTRACT

This final report summarizes Science Applications International Corporation's performance on contract no. N00014-86-C-2341 for the Naval Research Laboratory. Our principle deliverable, the codes RAM2D1 and PRAM1 have been completed on schedule and run successfully. Their operation is described in detail in this report as well as their application to cases of experimental importance. We have also carried out a number of analytical calculations in order to obtain greater insight into the code operation and the experiments and to make predictions in regimes of possible experimental interest which have not yet been explored. Some of these calculations were carried out in collaboration with Dr. John Reintjes of the Naval Research Laboratory. These calculations are summarized in this report. Relevant publications and presentations are also included.

I. INTRODUCTION

It is with pleasure and some pride that we present this summary of our accomplishments during this past year. We have completed the development of our principal deliverable, the code RAM2D1, which solves the basic equations governing transient, stimulated Raman interactions, accounting for both transient and diffractive effects. Using simple switches we can run the code in the transient regime where diffractive effects can be ignored or in the stationary regime where the transient effects can be ignored. Up to eight cases can be run simultaneously in these two limiting regimes. We have also developed a diagnostic code PRAM1 which uses DISSPLA routines to plot the results of our computer calculations. It can generate both ordinary plots and contour plots.

Both RAM2D1 and PRAM1 run on the NRL CRAY. They have also been tested on other CRAYs and should be easily modifiable to run on a variety of different machines.

In addition to developing and testing these codes, we have carried out a number of analytical and computational studies for the purpose of supporting existing experimental programs at the Naval Research Laboratory and exploring potential new ones. Some of these projects, but not all, make use of RAM2D1 and PRAM1. These studies have been marked by close cooperation with the experimentalists at the Naval Research Laboratory. We have explored transient phenomena in the long-distance limit both analytically and computationally. We have shown that the pump amplitude oscillates at a frequency proportional to $z^{1/2}$ and that the integrated intensity is proportional to $z^{-1/2}$ at long lengths. We have analytically studied stationary, multiple-beam interactions in a number of different limits. In collaboration with Dr. Reintjes of the Naval Research Laboratory, we have studied the conditions under which side beam replication occurs and have suggested a possible remedy. We have carried out computational studies of stationary, collinear beam propagation to determine the variation of the beam focal length due to nonlinear effects. In collaboration with Dr. M. Duncan and Dr. R. Mahon, we have also carried out transient, computational studies to make detailed comparisons between theory and experiment. Finally, we have carried out studies of solitons aimed toward determining when they will appear and whether they are worth studying experimentally.

These studies have laid a firm foundation for work which we will undertake in the years to come. The codes RAM2D1 and PRAM1 will undergo further development to enlarge the range of phenomena which can be considered, improve efficiency, and improve our diagnostic capabilities. Several of the scientific studies we have undertaken, particularly those concerning side beam replication and focal point evolution, are likely to remain major concerns in the coming years.

II. CODE DEVELOPMENT

A. Basic Philosophy

In this section, we outline the basic philosophy governing our choice of algorithms and our choice of the plotting package DISSPLA.

The basic equations which we need to solve are

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} = -i\kappa_2 \frac{k_L}{k_S} Q E_S \quad , \tag{2.1.a}$$

$$\frac{\partial E_S}{\partial_2} - \frac{i}{2k_S} \frac{\partial^2 E_S}{\partial y^2} = -i\kappa_2 Q^* E_L \quad , \tag{2.1.b}$$

$$\frac{\partial Q}{\partial t} + \Gamma Q = i\kappa_1 E_S^* E_L \quad , \tag{2.1.c}$$

where k_L, k_S, Γ, κ_1, and κ_2 are all physical parameters which are held constant in any individual computer run. Our boundary conditions are that $E_L(z,t)$ and $E_S(z,t)$ are fixed for all time at $z = 0$. We assume also that $Q(z,t) = 0$ at $t = -\infty$ for all z. Mathematically, Eqs. (2.1.a) and (2.1.b) are propagation equations while Eq. (2.1.c) is a constraint equation.

In solving these equations, our goal is to write a simple code which requires a minimum of space, runs with good efficiency, is robust, and is easily transportable.

The code RAM2D1 is written in FORTRAN and uses no canned routines except for the fast Fourier transform. Thus, this code is highly portable. The code PRAM1, being a plotting program, is dependent on the graphics package which is chosen. We use DISSPLA. While DISSPLA is somewhat difficult to learn, it is extremely powerful, and it exists on many different installations.

To solve the partial differential equations, we used a semi-spectral approach. For smooth initial conditions and infinite transverse boundaries, this approach has been shown for a large number of cases to be superior to finite difference or finite element methods.[1] The reason is that this approach is "infinite-order" in the transverse direction. It has the additional advantage that the linear propagation is solved exactly (to within computer roundoff) so that in the limit of weak nonlinearity, a quite important limit in practice, step sizes in z can remain relatively large.

Use of the semi-spectral method places a premium on carrying out the fast Fourier transform efficiently. We have written it so that it vectorizes in different directions in the fully two-dimensional and stationary limits. Other portions of the code are also written to vectorize as efficiently as possible.

Another concern is reducing memory requirements. For this reason, we settled on a mid-step Euler approach, rather than a fourth order Runge-Kutta approach, although the latter is more accurate.[2] We have found nonetheless that in the fully two-dimensional limit, the code is often too large to run on the CRAY-XMP32 at NRL without modification. We have thus written a version of the code which allows us to move the data back and forth from core memory to the disk, keeping only what is needed for a single operation in core. While this approach solves the space problem, it necessitates a substantial amount costly I/O. A completely acceptable solution to this problem has not yet been found.

A final issue that requires discussion is robustness. The semi-spectral approach with a mid-step Euler advancement in z is extremely robust. As long as sufficient spectral bandwidth is provided through a sufficient number of node points, the method is never linearly unstable. The other place this issue arises is in the solution of the constraint equation. One must integrate Eq. (2.1.c) in a way which yields an accurate solution, independent of the ratio of $T_{\max}$ to T_2, where $T_{\max}$ is the maximum $|t|$-value of the t-region being kept. In the region where $T_2 \gg T_{\max}$ we use straightforward integration. When $T_2 \ll T_{\max}$, we use a Fourier transform approach. The critical point is that each approach does not work well at the extreme limits of the regime opposite to where it is applied. Hence, both are needed. We have set the crossover point at $T_{\max}/T_2 = 10$. At the crossover point, both methods work well.

At present, the diagnostic code PRAM1 is set up to provide contour plots in $t-y$ space of the intensities, phases, and amplitudes of our principal fields, the pump, Stokes, and material excitation at fixed z-values. It also allows one to choose constant z and constant t sections for plotting purposes. These section plotting options are especially useful when plotting results obtained in the stationary and transient limits, as the contour plotting routines can no longer be used. The code PRAM1 does not plot z-history data; however, special, modified versions do exist for plotting this sort of data. More details can be found in the manual to RAM2D1 and PRAM1 which has been included as Appendix B.

The major concerns in designing this code have been flexibility and esthetics. Thus, the code has a large range of options, giving the user a large range of choices in how to plot the contour levels, where to choose the sections, how many to plot, and so on. Esthetic choices have included our insistence that the axis labels should be at "nice" values, the contour levels should be at "nice" values, and that the contour plots should lend themselves readily to the future production of movies.

B. Algorithmic Description of RAM2D1

The basic layout of RAM2D1 is shown in Fig. 2.1. In this section, we describe the basic algorithms used in each of the subroutines. The program listing is found in Appendix A, and more details on the variables and the program set-up can be found in Appendix B.

The main subroutine contains the input routine, routines to translate the input variables into variables used by the program, the basic stepping routine for the mid-step Euler method, and timing routines.

The variables `NT` and `NY` set the number of nodes in the t and y directions. These should be set equal to 2^N, where N is some integer, for the FFT routines to run properly.

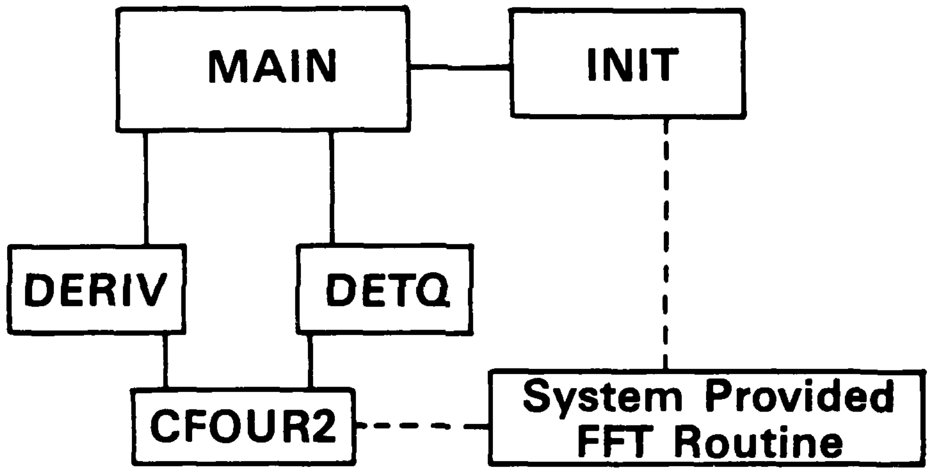


FIGURE 2.1. The structure of RAM2D1 is shown schematically.

Since they determine array sizes, they are set in a PARAMETER statement. The code must be re-compiled whenever they are changed. They also serve as switches. When NT$\leq$ 8, the code assumes that the stationary limit is being used with the number of distinct cases set by NT; similarly, when NY$\leq$ 8, the code assumes that the transient limit is being used with up to 8 distinct cases.

Most other input parameters are read in through a namelist statement. (The only exceptions are NP and NST which set the maximum number of pump beams and the length of the timing data vector.) Parameters include: the actual number of pump beams, NPUMP; parameters which specify the box size, TM and YM; parameters to determine beam offsets, intensities, and widths, YOFF, TOFF, YOST, TOST, YWIDTH, TWIDTH, YWST, TWST, RINT, RIST; basic switches used by INIT to set beam type, ICOND, RTYPE, ITYPE; other parameters governing the beam shape, RAMASM, RALASM, PHL, PHST, TOC, NHYP, RABAMP, RDSLIM; parameters governing the beam intersection point, the final z-value, and the z-step, ZINT, ZFINAL, ZSTEP; a parameter setting the maximum number of z-steps NMAX; physical parameters, RKP, RKS, TTWO, GAIN; and a parameter governing how often z-data is recorded, ZKEEP.

From here, initialization of the derived quantities is carried out. The parameters RKAP1 and RKAP2 (κ_1 and κ_2) are determined from GAIN (g). The pump and Stokes amplitudes are determined from the input power fluxes. Other miscellaneous actions are carried out; in particular the parameters governing the final z-value and the z-values at which data are recorded are reduced slightly to avoid difficulties with roundoff when making equality comparisons.

Since the step size is fixed, so is the linear propagator. To save computer time, we calculate the propagator once and for all and store it in CYVEC where it is called as needed. Two propagators are needed, one f . the pump and one for the Stokes. Specifically, referring to Eq. 2.A.1, and defining the Fourier transform, of $X(y)$

$$\tilde{X}(k) = \frac{1}{2\pi}\int_{-\infty}^{\infty} dy\, e^{iky} X(y) \quad , \tag{2.2}$$

we find that the linear propagator

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{i}{2k_L}\,\frac{\partial^2 E_L}{\partial y^2} &= 0 \quad , \\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S}\,\frac{\partial^2 E_S}{\partial y^2} &= 0 \quad , \end{aligned} \tag{2.3}$$

has the solution

$$\begin{aligned} \tilde{E}_L(k,z) &= \tilde{E}_L(k,0)\exp(-ik^2 z/2k_L) \quad , \\ \tilde{E}_S(k,z) &= \tilde{E}_S(k,0)\exp(-ik^2 z/2k_S) \quad . \end{aligned} \tag{2.4}$$

We note as well that the fast Fourier transform produces the k-values,

$$\begin{aligned} k &= \frac{2\pi(n-1)}{y_{\max} - y_{\min}} \quad , \qquad (1 \le n \le \mathtt{NY}/2) \\ k &= \frac{2\pi(n-1-\mathtt{NY})}{y_{\max} - y_{\min}} \quad , \qquad (\mathtt{NY}/2 < n \le \mathtt{NY}) \end{aligned} \tag{2.5}$$

where $y_{\min}$ and $y_{\max}$ are the minimum and maximum values of the y-box size.

We now initialize arrays which are used in the determination of Q. There are three methods used for determining Q, depending on the regime in which the code is being used. In method 1, which applies to the stationary limit, we set

$$Q = -i\kappa_1 \frac{E_S^* E_L}{\Gamma} \quad . \tag{2.6}$$

In method 2, which applies when $\mathtt{TRAT} = T_{\max}/T_2 < 10$, we use the integral expression

$$Q(z,t) = -i\kappa_1 e^{-\Gamma t}\int_{-\infty}^{t} e^{\Gamma t'} E_S^*(z,t') E_L(z,t')\, dt' \quad , \tag{2.7}$$

The integrand has no t-dependence which allows us to calculate the integral through a running application of the trapezoidal rule. The vector WQ1 is initialized to contain $\exp(\Gamma t)$, and the vector WQ2 is initialized to contain $\exp(-\Gamma t)$. Method 3 applies when TRAT>10. Here, we define the Fourier transform

$$\bar{X}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} dt\, e^{i\omega t} X(t) \quad . \tag{2.8}$$

We then note

$$\bar{Q}(z,\omega) = -i\kappa_1 \frac{\overline{E_S^*(z,\omega) E_L(z,\omega)}}{\Gamma - i\omega} \quad . \tag{2.9}$$

Our approach is thus to multiply E_S^* by E_L, take the Fourier transform, multiply by $-i\kappa_1/(\Gamma - i\omega)$ and inverse transform. The vector CWQ contains the Fourier data, and COMVEC contains the factor $-i\kappa_1/(\Gamma - i\omega)$. Method 2 does not succeed when TRAT becomes very large due to underflow/overflow problems. Method 3 does not succeed when TRAT becomes very small because $(\Gamma - i\omega)^{-1}$ becomes singular.

In the code, we next store our initial data and enter the routine INIT which initializes the pump and Stokes fields. The ICOND value determines the basic form of the initialization. When ICOND=1, a sech amplitude profile is taken in both the t and y directions. When ICOND=2, a sech^2 amplitude profile is taken in the t-direction and a hyper-Gaussian profile in the y-direction. Additional parameters adjust the leading edge of the Stokes pulse so that it can be sharper than the pump pulse and a chirp, or phase variation, can be added to the Stokes pulse. When ICOND=3, a transient case is run. The amplitude profiles are governed by ITYPE and include sech, Lorentzian, square, exponential, Gaussian, and hyper-Gaussian profiles. The power of the amplitude or exponent is governed by RTYPE. When ICOND=4, a stationary case is run with a hyper-Gaussian profile. Other parameters allow non-zero phase and intensity aberrations to be introduced and govern their strength.

The subroutine INIT functions as a separate element and takes up increasing space as more alternative initializations are added. At present, however, the space allocated is negligible, and it is not worthwhile to separate INIT from RAM2D1.

We next record the initial fields and, except in the transient case where Fourier-transformed data is not used, calculate and record the Fourier data as well.

We now enter the main loop where the mid-step Euler method is applied to the forward stepping of $E_L(z,y,t)$, $E_S(z,y,t)$, and $Q(z,y,t)$ from z_n to $z_{n+1} = z_n + \Delta z$. We assume that E_L, E_S, and Q are known at $z = z_n$. We recall that the equations for $\tilde{E}_L$ and $\tilde{E}_S$ are

$$\begin{aligned} \frac{\partial \tilde{E}_L}{\partial z} + \frac{ik^2}{2k_L}\tilde{E}_L &= -i\kappa_2 \frac{k_L}{k_S}\widetilde{QE_S} \quad , \\ \frac{\partial \tilde{E}_S}{\partial z} + \frac{ik^2}{2k_S}\tilde{E}_S &= -i\kappa_2 \widetilde{Q^* E_L} \quad . \end{aligned} \tag{2.10}$$

The routine DERIV calculates the right-hand sides of Eq. (2.10) by first determining the multiplicands QE_S and Q^*E_L and then transforming. The quantities $\tilde{E}_L$ and $\tilde{E}_S$ are then advanced to the midpoint using the formula

$$\begin{aligned} \tilde{E}_L(z_{n+1/2}) &= \exp\left[(-ik^2/2k_L)(\Delta z/2)\right]\tilde{E}_L(z_n) \\ &\qquad - i\frac{\Delta z}{2}\kappa_2\frac{k_L}{k_S}Q(z_n)E_S(z_n) \quad , \\ \tilde{E}_S(z_{n+1/2}) &= \exp\left[(-ik^2/2k_S)(\Delta z/2)\right]\tilde{E}_S(z_n) \\ &\qquad - i\frac{\Delta z}{2}\kappa_2 Q^*(z_n)E_L(z_n) \quad , \end{aligned} \tag{2.11}$$

where $z_{n+1/2} = z_n + \Delta z/2$. We recall that the exponential factors are stored in CYVEC. The routine DETQ first determines $E_L(z_{n+1/2})$ and $E_S(z_{n+1/2})$ by inverse transforming $\tilde{E}_L$ and $\tilde{E}_S$. It then calculates $Q(z_{n+1/2})$ using the appropriate method. We now repeat the procedure, first using DERIV to calculate the right-hand side of Eq. (2.10) at $z = z_{n+1/2}$ and then using the formula

$$\begin{aligned} \tilde{E}_L(z_{n+1}) &= \exp\left[(-ik^2/2k_L)\Delta z\right]\tilde{E}_L(z_n) \\ &\qquad - i\Delta z\,\kappa_2\frac{k_L}{k_S}Q(z_{n+1/2})E_S(z_{n+1/2}) \quad , \\ \tilde{E}_S(z_{n+1}) &= \exp\left[(-ik^2/2k_S)\Delta z\right]\tilde{E}_S(z_n) \\ &\qquad - i\Delta z\,\kappa_2 Q^*(z_{n+1/2})E_L(z_{n+1/2}) \quad , \end{aligned} \tag{2.12}$$

to determine $\tilde{E}_L$ and $\tilde{E}_S$ at $z = z_{n+1}$. Using DETQ, we finally obtain E_L, E_S, and Q at $z = z_{n+1}$ and are ready for the next loop iteration. For transient runs, Fourier transform data is not calculated.

At the end of each loop iteration, a check is made to determine whether data should be recorded.

After exiting the main loop, the timing data and the number of data records is recorded.

C. Algorithmic Description of PRAM1

The purpose of PRAM1 is to display desired aspects of the data that RAM2D1 generates and files. Those data are the complex field amplitudes of the pump beams, Stokes beam, the material excitation, and their Fourier transform representations (=6 field arrays). Whenever Fourier transforms are mentioned it is understood to be the transform with regard to the transverse spatial dimension y. The desired format of display is that of contour plots of the intensity of these fields versus time coordinate and versus transverse spatial coordinate at a given point z along the path of propagation. In addition to that, cross-sections (of the contour plots) of the intensity and sections of the field phase and amplitude at user-defined z-values can be displayed. In the case of one-dimensional simulations no contour plots are available.

Intensity, phase, and real and the imaginary part of the amplitude can all be diplayed. These three types of plots are desired of the three fields and their Fourier transforms. Hence, 18 different types of sections in addition to the intensity contour plots can be generated.

The user can generate any one or several graphs of the described type by specifying appropriate values for the elements of the flagging vector ISRF and array CSEC in the input data file NP-.DAT. For this purpose the field arrays are numbered (I through VI) and the sections (1-18). Which numeral corresponds to which type of graph and their sequence of appearance in the output is as follows:

I. contour plot of pump intensity

1. sections of pump intensity
2. sections of pump phase
3. sections of pump amplitude (real/imag)

II. contour plot of pump FFT intensity

4. sections of pump FFT intensity
5. sections of pump FFT phase
6. sections of pump FFT amplitude (real/imag)

III. contour plot of Stokes intensity

7. sections of Stokes intensity
8. sections of Stokes phase
9. sections of Stokes amplitude (real/imag)

IV. contour plot of Stokes FFT intensity

10. sections of Stokes FFT intensity
11. sections of Stokes FFT phase
12. sections of Stokes FFT amplitude (real/imag)

V. contour plot of mat. exct. intensity

13. sections of mat. exct. intensity
14. sections of mat. exct. phase
15. sections of mat. exct. amplitude (real/imag)

VI. contour plot of mat. exct. FFT intensity

16. sections of mat. exct. FFT intensity
17. sections of mat. exct. FFT phase
18. sections of mat. exct. FFT amplitude (real/imag)

VII. contour plot of pump and Stokes intensity

19. sections of sum of pump and Stokes intensity

VIII. contour plot of pump and Stokes FFT intensity

Three more types of plots than the expected 6+18 were added on the bottom of the list. These are the surfaces VII and VIII and section 19. These graphs plot pump and Stokes data simultaneously on a common scale. Section 19 draws the sum of the fields weighted by a certain factor so as to compose a special invariant of the Raman interaction.

The roman numerals tell which element of the vector ISRF is the flag that determines if that particular contour plot will be generated or skipped.

ISRF(n) $= 0$ contour plot skipped

ISRF(n) $= 1$ contour plot drawn with contours labeled

ISRF(n) $= -1$ contour plot drawn; no labels on contours

The default value is zero which is set in PRAM1.

Each sectional plot is associated with one element of the complex array CSEC. The position of the element specifies uniquely one cross sectional plot. The arabic numerals in the list above indicate the row number (first array index) of CSEC with which each described type of section is associated. The column number (second index) of the elements of CSEC numbers the particular cross sectional plot of that type. The parameter NSEC (≤ 9) gives the maximally allowed number of sections of each type for the run.

The plotting of any cross-section is done depending on the values of the real and imaginary part of their representative element in CSEC. At the beginning of execution PRAM1 sets them all equal to zero as the default value. The values specified in the input data file replace these zeros.

In a two-dimensional simulation the value of the imaginary part of each element of the array CSEC means:

= 0.0: that this sectional plot is not requested

= 1.0: that this sectional plot is requested and that it shall be a cross-section parallel to the y-axis of the surface under question at a fixed t-value as given in physical units (psec)by the real part of the current element of CSEC. The first index of the array(s) SRF(I) in PRAM1 is being held constant for this plot at the value ISEC which is the grid point that corresponds best to the fixed t-value;

= 2.0: that this sectional plot is requested and that it shall be a cross section parallel to the t-axis of the surface under question at a fixed y-value as given in physical units (cm or 1/cm)by the real part of the element of CSEC in question. The second index of the array(s) SRF(I) in PRAM1 is being held constant for this plot at the value ISEC which is the grid point that corresponds best to the fixed y-value.

In short: the imaginary part tells which variable to hold constant, and the real part tells at what value (in physical units).

In a one-dimensional simulations the location of the array elements within CSEC has the same correspondence with cross-sectional plots as in two-dimensional simulations. The exact values of the imaginary parts of CSEC no longer matter except that they must be larger than 0.001 for the corresponding sections to be generated. In the diagnosis of a run by RAM2D1 where only one one-dimensional case was simulated the exact value of the real parts of CSEC also do not matter; however, they must be larger than 0.5 and less than 8.5 for the plot to be generated. When several cases have been simulated in one run simultaneously the value of the real part (1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, or 8.0) of each element of array CSEC identifies which of these cases is meant to be diagnosed.

The program has the structure shown in Fig. 2.2. The program starts by setting default values for the graphics output parameters as specified in the data statements. These default values are updated by reading customized values from the namelist file NPRAM1 The updated parameter set is then written, depending on the value of the 4 elements of the vector LPRMT(n), onto the first 4 graphics frames in the output META-file call PLT2.DAT. The value of the n-th element of LPRMT should be equal to 1 if the n-th page of parameters is to be plotted, and equal to 0 if not. The content of the four parameter graphics frames is shown in the examples of the appendices.

The program continues by calculating several constants. Among these constants are the end values and interval sizes for the plotting of the frequently used *y*- and *t*-coordinate axes. Then the large DO-loop 500 is entered. It reads the data for each requested plot and converts it into a device-independent graphics frame that is stored in the META-file. Once all data are scanned with respect to the requested graphs the generated graphics frames in the META-file are transfered to the VAX storage disk under the name PLT2.DAT.

In detail, the ensuing main part of this program acquires the electric field data from the input data file F——— by reading sequentially the i-th record specified by the value i of the consecutive elements of the vector KZ. These complex amplitude data are converted into real intensity data (array SRF), or are split into their real (array SRF) and imaginary (array SRFI) parts for plotting of the phase and/or amplitudes. Following their accquisition, the real arrays, SRF and SRFI, are handed, like other necessary parameters, through FORTRAN COMMON BLOCKS to the subroutine CNTR (for contour plotting) and to the subroutine CRSSCT (for cross sectional plots).

The subroutine CNTR is then called depending on the value of the relevant element of ISRF. Interleaved in these calls are the calls to the subroutine CRSSCT, depending on the sum of values of all elements of the line of CSEC in question. If this sum is non-zero CRSSCT is called to generate the requested graphs, if this sum is zero the DO-loop proceeds. Following the end of DO-loop 500 the program just closes the META-file and then stops execution.

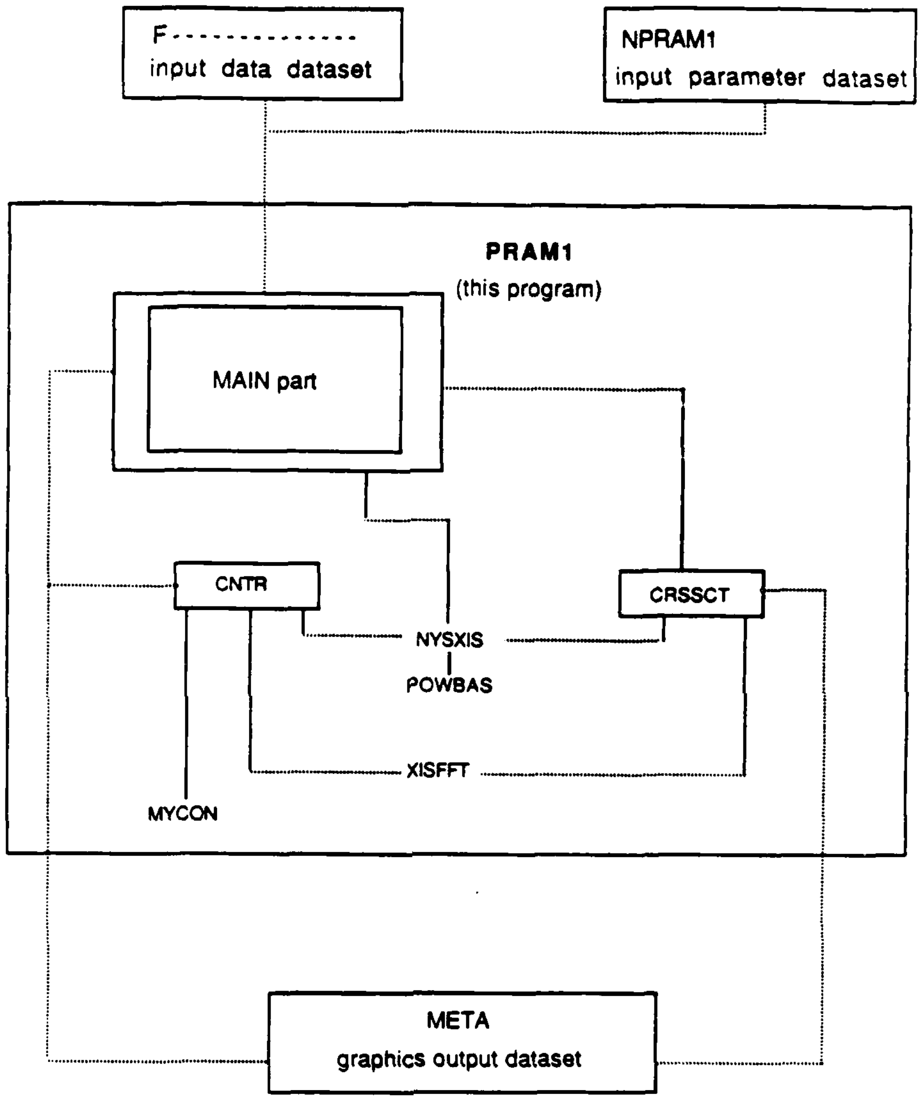


FIGURE 2.2. The structure of PRAM1 is shown schematically.

The contouring subroutine CNTR makes use of the subroutine MYCON which generates a customized dotted line for the half-height contour. The cross section subroutine CRSSCT calls repeatedly on the subroutine NYSXIS which finds "nice" values for coordinate axis limits and intervals. NYSXIS in turn uses subroutine POWBAS to find the next lower integral power of 10 for maximas and minimas. Both subroutines CNTR and CRSSCT share subroutine XISFFT when making secondary axes for FFT-plots.

The subroutine CNTR contains the calls to the special routines of the DISSPLA graphics library that create the graphics data for each contour plot. Several features of those plots are customized with respect to the standard that DISSPLA provides. For example, the software that generates axis tick marks has been amended by the non-DISSPLA subroutine NYSXIS. The subroutine NYSXIS computes "nice" tick marks along the coordinate axes. The subroutine XISFFT computes the location, extrema and intervals of the transformed variable axis in FFT-plots. The subroutine changes the field data in order to plot the logarithmic intensity as desired. Therefore, the intensity data in SRF (SRFI) have to be restored in the main part of the program before the intensity cross sections can be generated.

The call to subroutine CNTR contains the parameter KSRF which identifies the type of graph. Depending on its values, various titles, coordinate axes, and labels are selected and drawn. The sign of KSRF toggles the labeling option of the main contour lines (positive KSRF labels, negative KSRF no labels). The main contour lines are solid lines representing integral powers of 10. A number NDEC such lines will be drawn below the peak of the surface maximum. A number ILN (≤ 8) other contour lines (dashed lines) are drawn between the main contour lines corresponding to the integral multiples of the next lower integral power of ten. Which integral multiples are to be drawn is determined by the first ILN elements of the vector LEVEL. If the input parameter ISHM $= 1$ a dotted contour will mark the half-height level, if ISHM $= 0$ this line will not be drawn, if ISHM $= -1$ the half-height contour and a dot at the surface maximum will be drawn.

The subroutine CRSSCT contains the calls to the special routines of the DISSPLA graphics library that create the graphics data for each cross sectional plot from the modified field data in the array(s) SRF (SRFI). Just as in the case of CNTR, several features of those plots are customized with respect to the standard that DISSPLA provides.

The three categories of cross sectional plots are: intensity plots (following statement label 300), phase plots, and amplitude plots (both following statement label 400). When intensity cross sections are called for, this subroutine executes DO-loop 390 that does all cross sections specified in row MSRF of array CSEC and thereafter returns control to the main program. When phase or amplitude cross sections are called for, this subroutine executes DO-loop 490 which generates all phase sections specified in row MSRF of array CSEC. Immediately afterwards, since phase plots and amplitude plots are derived from the same data in the arrays SRF and SRFI, DO-loop 590 is executed which generates all amplitude cross sections that are specified in row MSRF+1 of array CSEC. After these actions, control is returned to the main program.

Each type of cross sections is prepared in a similar fashion. In the case of one-dimensional data (NT or NY less than or equal to 8), only one argument of the array(s) SRF (and SRFI) is an independent variable the other argument serves as a label to allow distinction between up to eight one-dimensional datasets. Which one of these eight datasets is to be graphed is determined by the value of the real part of the element of CSEC under consideration. When NT and NY are larger than 8, then SRF and SRFI contain one two-dimensional function, a surface. Which of the two functional arguments is to be held constant for each cross sectional plots is determined by the imaginary part of its corresponding element of CSEC. Therefore, in 2-d cases the imaginary part of the current element of CSEC is tested. If it is 2.0 a horizontal cross section (second variable of array(s) SRF (SRFI) fixed) follows; if it is 1.0 a vertical cross section (first variable of array(s) SRF (SRFI) fixed) follows; otherwise the next element in the current row of CSEC will be considered in the same way. A selected plot starts by writing its headline and axis labels onto a new graphics frame. Then the data of the sectional curve are computed, the coordinate system is sized accordingly and then drawn. Finally the cross sectional curve is itself drawn. If the plot displays FFT-data the drawing of the FFT-axis that would be drawn as part of the coordinate system (CALL GRAF) will be suppressed in order to avoid the tick mark labels which generally exhibit "messy looking" numbers. This axis of the coordinate system is suppressed. Instead of it a "secondary" (DISSPLA nomenclature) axis will be drawn immediately after the cross sectional curve is drawn. This secondary axis exhibits tick marks with "nice" values as determined by the subroutine NYSXIS.

The cross sectional curves are the functional values of the field data arrays at the grid point ISEC that is the closest to the locations specified by the real part of the current element of CSEC. While the data of the intensity and amplitude can readily be ploted as they are available in the array(s) SRF (SRFI), the data for the phase sections have to be calculated first by this subroutine.

The plotted phase data are calculated as follows: The field magnitude at the fixed grid point ISEC is computed. If its maximum is less than 10^{-30} the field information is deemed unreliable and no phase curve will be drawn. Furthermore all locations where the magnitude is less than the maximum magnitude divided by 10^{8} are deemed unreliable and no phase curve points are shown. The arctangent of the ratio of the imaginary to real field amplitudes provides the raw phase data. It is assumed that the numerical resolution of RAM2D1 is sufficient to provide raw phase data that do not vary by more than $\pm\pi$ from grid point to grid point. The first raw data point is placed within $\pm\pi$ of zero phase. All consecutive raw data points are tested if they were reached by a phase change that implies a crossing of the negative real axis of the amplitude vector in which case 2π will be added or subtracted to all following phase points depending on an implied phase wind-up or wind-down. By this method phase variations crossing multiple 2π-intervals can be followed. In case of intermittent unreliable data points the next reliable phase is placed within the 2π-interval of the previous reliable phase point.

The subroutine NYSXIS finds "nice" end values and interval step sizes (used for customized axis labeling) outside of the range that is specified by a choice of two from the following four values in the subroutine arguments: the maximum value in the vector VEC, the minimum value in VEC, VECBOT, and VECTOP. The decision which quantities constitute the reference interval depends on the value of the argument NECLEC.

If NECLEC $= -1$: VECBOT and VECTOP are chosen and the vector VEC is neglected.

$= 0$: then the maximum and minimum of all four are chosen

$= 1$: then the extrema of VEC are chosen and VECBOT and VECTOP are neglected. It is also possible to "hard-wire" the lower (upper) end-value to the current value of VECBOT (VECTOP) by setting the argument VECGAP to -1.0 (1.0) as input. If VECGAP $= 2.0$ on input both end values are "hard-wired."

The subroutine NYSXIS finds the extrema of the input data. Then it determines the largest integral power of ten (XTRPOW) that is still smaller than the larger of the absolute values of the extrema. Based on XTRPOW the leading two decimal places of the extrema are compared with each other. The possible difference is placed into one of seven interval classes with the following interval sizes: 0.005, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0 times XTRPOW. The end values that will be returned are chosen to be one interval beyond the integer that is closest to the original extrema. If the hard-wiring option was chosen the hard-wired end value is re- instated before the interval and end values are returned to the calling routine.

III. SCIENTIFIC STUDIES

III.A. Transient Raman Interactions

The study of transient Raman effects has been an important focus of scientific activity since the original papers of Wang[3] and of Carmen, *et al.*[4] Recent experiments at the Naval Research Laboratory[5] have reinvigorated basic research in this area and opened up new questions relating to the evolution of pulses in this regime. The basic equations are

$$\begin{aligned} \frac{\partial E_L}{\partial z} &= -i\frac{k_L}{k_S}\kappa_2 Q E_S \ , \\ \frac{\partial E_S}{\partial z} &= -i\kappa_2 Q^* E_L \ , \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i\kappa_1 E_S^* E_L \ , \end{aligned} \tag{3.1}$$

where E_L and E_S are the pump and Stokes fields, z and t are axial distance along the Raman interaction cell and time with z-dependent origin, k_L and k_S are the pump and Stokes wavenumbers, and κ_1 and κ_2 are Raman coefficients.

We have carried out analytical and numerical calculations, both to gain insight into behavior that has been observed experimentally at NRL and to predict the behavior that would be observed in regimes not yet accessed by the experiments. Virtually all the analytical work is summarized in the preprint "Asymptotic evolution of transient pulses undergoing stimulated Raman scattering," which is included in Appendix C of this report and will not be repeated in detail here. Basically, we have shown that if the amplitude of the initial Stokes is small compared to the amplitude of the initial pump, then the pulse evolution passes through two main regimes. Initially, the Stokes grows exponentially while the pump is essentially undepleted. During this growth, the phase of the Stokes pulse locks onto the pump phase. This regime was studied by Carmen, *et al.*[4] using simple linear theory. We call it the I-regime. This regime is followed by a transition regime in which pump depletion becomes significant. Finally, there is the J-regime in which the Stokes intensity remain almost constant while the pump slowly depletes.

We now turn to our computational studies, considering first the J-regime. In Figs. 3.1–3.2, we show the variation of

$$R = \left[\int_{-\infty}^{\infty} dt\,|E_L|^2(0) / \int_{-\infty}^{\infty} dt\,|E_L|^2(\varsigma)\right]^2$$

vs. $\varsigma = \kappa_1\kappa_2 z \int_{-\infty}^{\infty} K(t)\,dt$, where

$$K(t) = |E_L|^2(\varsigma,t) + \frac{k_L}{k_S}|E_S|^2(\varsigma,t)\ . \tag{3.2}$$

The theory indicates that R should vary linearly with ς at sufficiently large ς. This trend is observed in Fig. 3.1. Moreover, linear behavior is observed when $R \gtrsim 10$ which corresponds to approximately 70% depletion of the pump. In Fig. 3.2, we show on a parabolic scale vs. ς the number of zero-crossings N of the pump amplitude. Theory indicates that N^2 should be proportional to ς. This result is confirmed in Fig. 3.2. We note that in all cases the expected asymptotic behavior is observed for $\varsigma \gtrsim 120$, and the original Stokes has an intensity 0.001 that of the pump.

In Fig. (3.3) and (3.4), we show the effect of varying the Stokes offset for pulses with an initial sech^2 amplitude and a FWHM of 40 ps. At negative offsets there is a tendency for depletion to be delayed while the number of zero-crossings increases linearly beyond a relatively short distance. When $t_{\rm off} = -40$ ps, the pump must be 90% depleted before linear variation of R is observed. At $t_{\rm off} = -20$ ps, one finds that R begins to scale linearly when the pump is 85% depleted. At $t_{\rm off} \geq 0$ ps, this requirement reduces to 70% depletion. In all cases, linear behavior is observed to set in when $\varsigma = 100$–200, with the highest vlaues occuring at the most negative offsets. Conversely, when $t_{\rm off} > 0$, there is a tendency for the oscillations of the pump amplitude to be delayed. If we add the effect of chirp onto the pump pulses, there is little effect until the chirp becomes quite sizable. With a phase

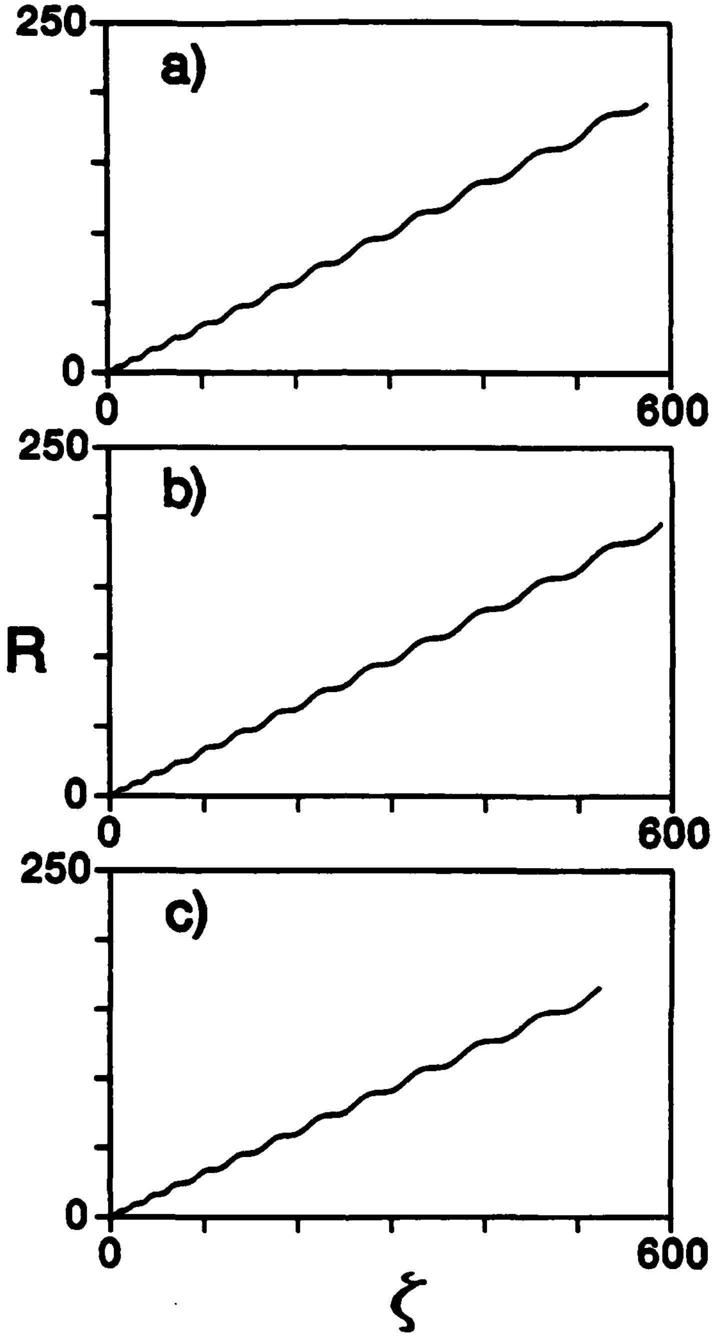


FIGURE 3.1. Plots of R vs. ς for different pulse shapes. a) sech-squared amplitude, FWHM = 40 ps; b) Lorentzian-squared amplitude, FWHM = 39 ps; c) Square pulse, FWHM = 43.8 ps.

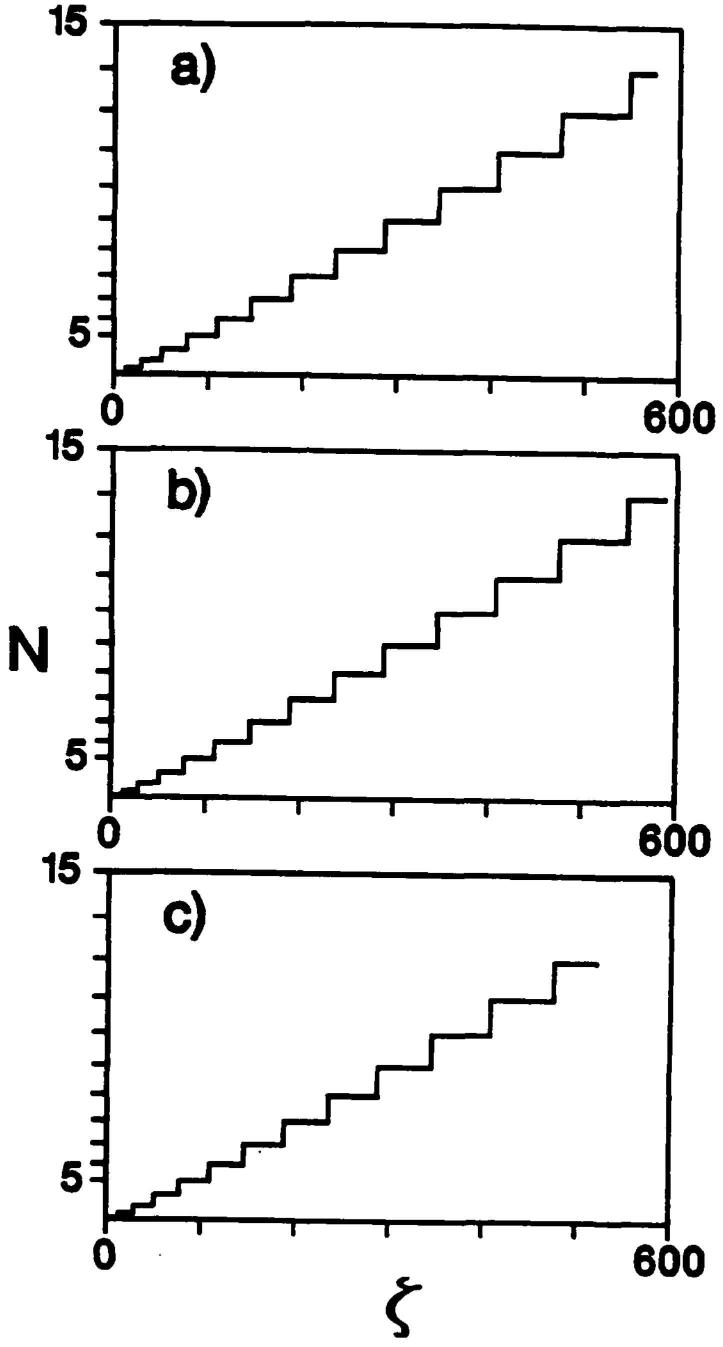


FIGURE 3.2. Plots of N vs. ς; N is plotted on a parabolic scale. Shapes and parameters are the same as in Fig. 3.1.

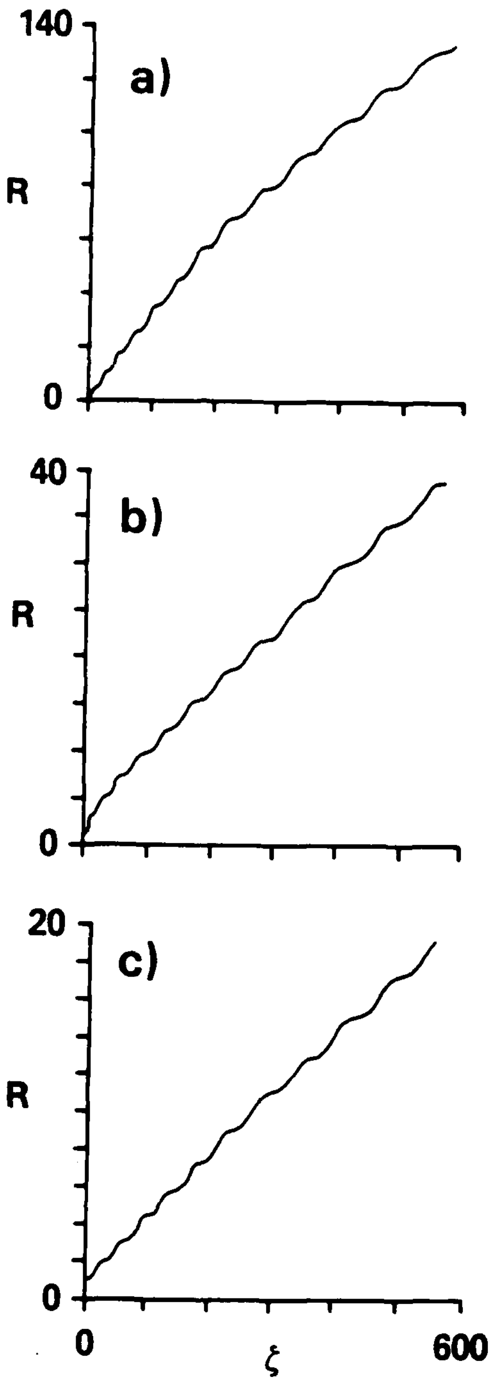


FIGURE 3.3. Effect of Stokes offset on the scaling of R with ς. In all cases the pulses have a sech^2 amplitude profile and a FWHM of 40 ps. a) $t_{\text{off}} = -20$ ps; b) $t_{\text{off}} = 0$ ps; c) $t_{\text{off}} = 20$ ps.

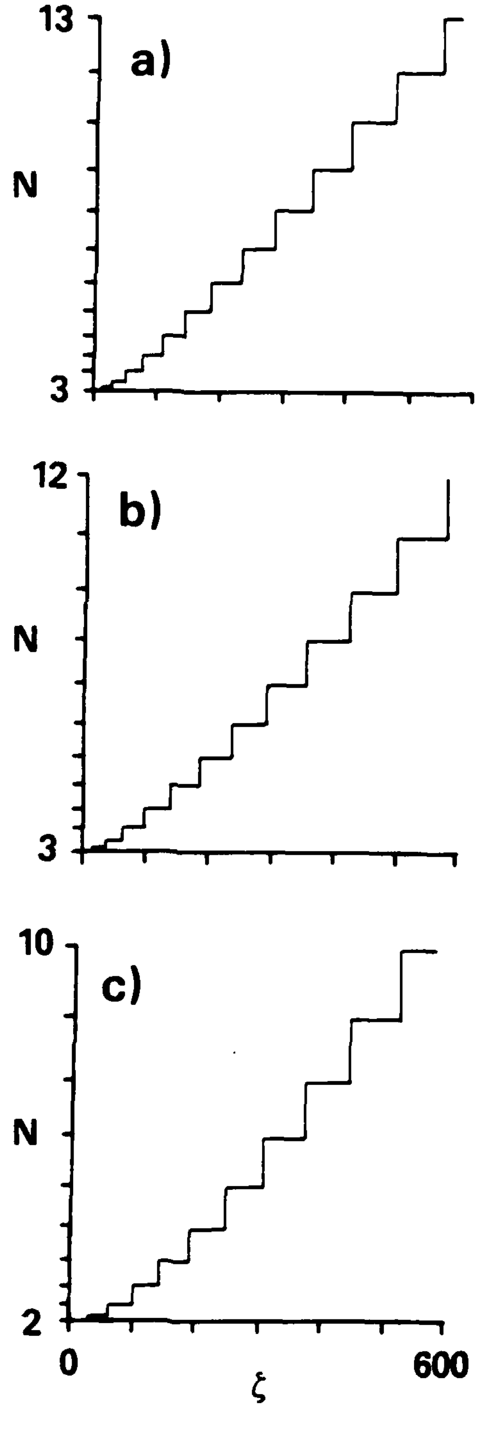


FIGURE 3.4 Effect of Stokes offset on the scaling of N with ζ. Parameters are the same as in Figure 3.3.

variation of 9.5π, we find an increase by under 50 in the ς-values at which linear behavior of R sets in. Otherwise, the qualitative behavior remains the same.

Turning now to consideration of the I-regime, we consider the effect of the Stokes pulse offset on its gain over a fixed distance (40 cm) and its ability to phase lock to the pump. For a symmetric pulse with an initial sech^2 amplitude, a 40 ps FWHM, an initial maximum pump intensity of 1.0 Gwatts/cm^2, and an initial Stokes intensity 0.001 the pump intensity, we list the dependence of gain and locking on offset in Table III.1. The chirp referred to is approximately π, which is the experimental magnitude. In Fig. 3.5, we show the phases at $z = 40$ cm for three values of $t_{\rm off}$. Phase locking is complete at $t_{\rm off} = -20$ ps and $t_{\rm off} = 0$ cm but is incomplete at $t_{\rm off} = 0$ cm.

Finally, we have carried out numerical calculations aimed at understanding the rapid phase flip which is observed experimentally to travel from the back to the front of the pulse in the I-regime. We generally observe from our numerical studies that a fast phase flip can be obtained at a particular gain for a fixed chirp. As we raise the gain, this fast phase flip disappears and phase locking occurs in contrast to the experimental observations. It appears at this point that we will have to take into account diffractive effects in order to have any hope of explaining the experimental observations, and we will carry out those studies shortly.

III.B. Beam Interactions in the Stationary Limit

In this section we outline a number of analytical calculations which we have made in the stationary limit to clarify the effect of aberrations and a finite interaction length on the interaction of pump beams with a Stokes beam in both collinear and crossing beam geometries.

In studying a multiple beam geometry, we may assume that the pump beam has the

TABLE III.1

Variation of Gain and Locking With Offset

	No Chirp	Chirp	
Offset	Gain	Gain	Locking
−100	1.0	1.0	locked
−75	2.5	1.25	locked
−50	15	5.8	locked
−25	42	15	locked
0	43	23	locked
25	17	11	partially locked
50	2.7	1.4	none
75	1.1	1.1	none

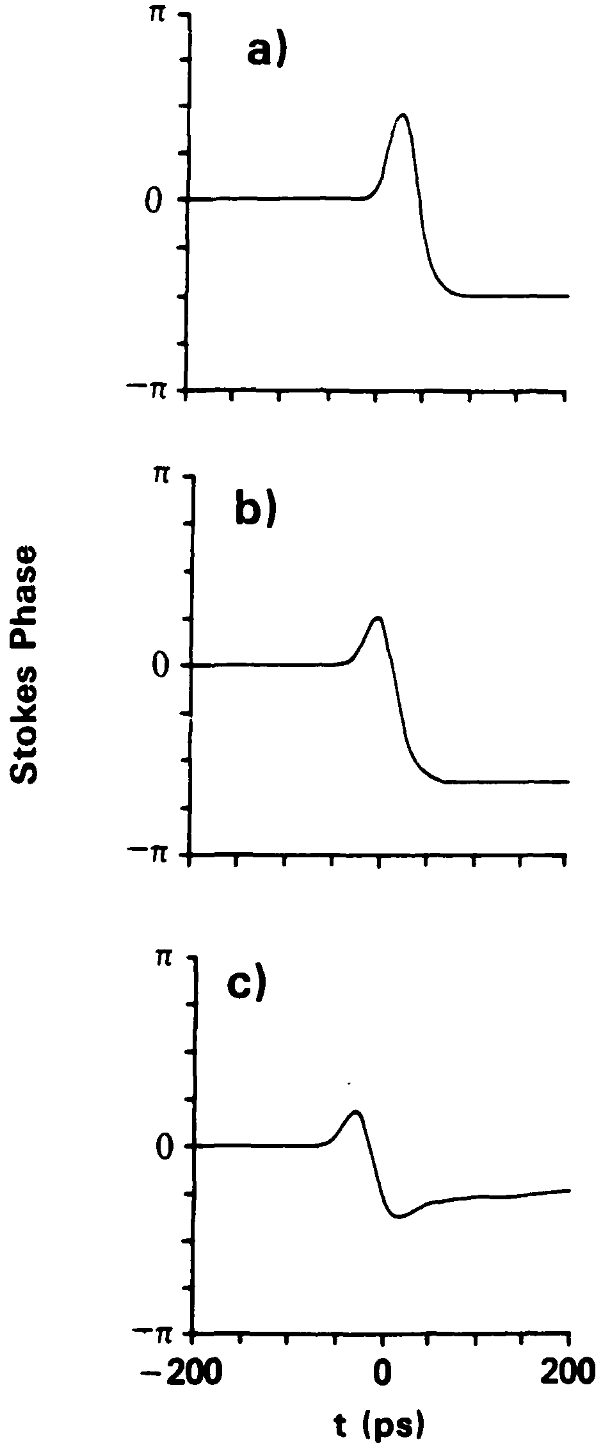


FIGURE 3.5. Phase-locking of a pulse with sech^2 amplitude is shown at three different offsets. a) $t_{\text{off}} = -20$ ps; b) $t_{\text{off}} = 0$ ps; c) $t_{\text{off}} = 20$ ps.

form

$$E_L = \sum_{n=N}^{N} f_n\left[y + (ny_0/z_0)z - ny_0, z\right] \exp\left\{-ik_L(ny_0/z_0)\left[y + \frac{1}{2}(ny_0/z_0)z\right]\right\} \quad , \tag{3.3}$$

where the f_n give the shapes of the individual beams. The first argument gives the rapid transverse variation. The second argument gives the slow z-variation due to diffraction. The quantity y_0 gives the intrinsic separation between the beams when $z = 0$ while z_0 is the z-value at which all the beams converge. Noting that

$$\tilde{X}(k) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(y) \exp(-iky)\, dy \quad , \tag{3.4}$$

we find

$$\begin{aligned} \tilde{E}_L(k) = & \sum_{n=-N}^{N} \tilde{f}_n[k = k_L(ny_0/z_0), z] \\ & \exp\left\{i\left[k + k_L(ny_0/z_0)\right]\left[(ny_0/2z_o)z - ny_o\right]\right\} \quad . \end{aligned} \tag{3.5}$$

In the Fourier domain, $\tilde{E}_L$ consists of a set of offset peaks. In all experiments, the width of these peaks $(\Delta K)_{\text{beam}}$ can be assumed small compared to the fundamental separation between the peaks $(\Delta K)_{\text{sep}} = y_0/z_0$. When the beams are well-separated in the coordinate domain, they have rapid amplitude modulations in the Fourier domain; when the beams are nearly overlapping, these amplitude modulations are slow.

We note that the condition $(\Delta K)_{\text{beam}} \ll (\Delta K)_{\text{sep}}$ results in the nonlinear terms combining like convolutions. The basic equations reduce in the stationary limit to

$$\begin{aligned} \frac{\partial E_L}{\partial z} - \frac{i}{2k_L} \frac{\partial^2 E_L}{\partial y^2} &= -\frac{g}{2} \frac{k_L}{k_S} |E_S|^2 E_L \quad , \\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S} \frac{\partial^2 E_S}{\partial y^2} &= \frac{g}{2} |E_L|^2 E_S \quad . \end{aligned} \tag{3.6}$$

We now find, letting $y_n = y + (ny_0/z_0)z - ny_0$, $z_n = z$, and

$$E_S = \sum_{n=-N}^{N} g_n(y_n, z_n) \exp\left\{-ik_S(ny_0/z_0)\left[y + \frac{1}{2}(ny_0/z_0)z\right]\right\} \quad , \tag{3.7}$$

that

$$\frac{\partial g_n}{\partial z_n} = \sum_{k=-N}^{N}\sum_{l=-N}^{N}\sum_{m=-N}^{N} f_k(y_n,z_n) f_l^*(y_n,z_n) g_m(y_n,z_n)$$
$$\cdot \exp\left\{-ik_L\left[\left(\frac{ky_0}{z_0}\right)^2 - \left(\frac{ly_0}{z_0}\right)^2\right]z - ik_S\left[\left(\frac{my_0}{z_0}\right)^2 - \left(\frac{ny_0}{z_0}\right)^2\right]z\right\} \quad , \tag{3.8}$$

where we impose the condition $k - l + m - n = 0$. Terms which do not satisfy this condition are too rapidly varying in y to be consistent with the condition $(\Delta K)_{\text{sep}} \ll (\Delta K)_{\text{beam}}$.

In general, the exponential factor in Eq. (3.8) is also rapidly varying. Exceptions are when $k^2 = l^2$ and $m^2 = n^2$, in which case this factor disappears. It turns out that only these cases make non-zero contributions to dg_n/dz_n. This issue is discussed in great detail in the paper "Pump replication in stimulated Raman scattering using a crossed-beam geometry" which we presented at the 1988 SPIE meeting in Los Angeles in session 874. This paper is included in Appendix C, and we do not present the details here. This paper has a discussion of the effect of pump aberrations on side beam replication which we also do not repeat.

At this point we discuss the effect of geometry on Stokes beam amplification in the low Fresnel number regime where diffraction can be ignored. We consider as a simple example two crossing beams with Gaussian profiles interacting with a single Stokes. Thus, $k = -l = \pm 1$ and $m = n = 0$ in Eq. (3.8). Writing

$$f_1 = \frac{E_0}{(2\pi)^{1/2}w}\exp\left\{-\left[y - (y_0/z_0)z + y_0\right]^2/2w^2\right\} \quad ,$$
$$f_{-1} = \frac{E_0}{(2\pi)^{1/2}w}\exp\left\{-\left[y + (y_0/z_0)z - y_0\right]^2/2w^2\right\} \quad , \tag{3.9}$$

we conclude that

$$\frac{dg_0}{dz} = \frac{E_0^2}{2\pi w^2}\Big(\exp\left\{-\left[y - (y_0/z_0)z + y_0\right]^2/w^2\right\}$$
$$+ \exp\left\{-\left[y + (y_0/z_0)z - y_0\right]^2/w^2\right\}\Big)g_0 \quad . \tag{3.10}$$

Integrating Eq. (3.10), we find

$$g_0(y,z) = g_0(y,0)\exp\left\{\frac{z_0 g E_0^2}{8\sqrt{\pi}y_0 w^2}\,\mathrm{erf}\left(\frac{y_0 z}{z_0 w} + \frac{y-y_0}{w}\right) + \mathrm{erf}\left(\frac{y_0 z}{z_0 w} - \frac{y+y_0}{w}\right) - \mathrm{erf}\left(\frac{y-y_0}{w}\right) + \mathrm{erf}\left(\frac{y+y_0}{w}\right)\right\} \quad . \tag{3.11}$$

The maximum gain which is achieved in the limit $y_0 \gg w$ and $z \gtrsim 2z_0$ is

$$g_0(y,z) = g_0(y,0)\exp\left(\frac{g z_0 E_0^2}{2\sqrt{\pi}y_0 w^2}\right) \quad . \tag{3.12}$$

The gain saturates because the pump and Stokes beams only interact over a limited length.

When a pump beam is aberrated, its linear propagation is strongly affected. Suppose we consider a Gaussian beam with phase aberrations

$$E_L(z=0) = \frac{E_0}{\sqrt{2\pi w}}\exp(-y^2/2w)\exp[i\varphi(y)] \quad , \tag{3.13}$$

were $\varphi(y)$ is a randomly varying phase. Then E_L for $z > 0$ can be determined by solving the linear equation

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_L}\ \frac{\partial^2 E_L}{\partial y^2} = 0 \quad . \tag{3.14}$$

We find that

$$\begin{aligned}|E_L(z,y)|^2 = \frac{1}{(2\pi)^2}\int_{-\infty}^{\infty} dk \int_{-\infty}^{\infty} dk' \int_{-\infty}^{\infty} dy' \int_{-\infty}^{\infty} dy'' \left(\frac{E_0^2}{2\pi w^2}\right)& \\ \exp\left[i(k-k')y\right]\exp\left[-iky' + ik'y''\right]& \\ \exp\left\{-\left[(y')^2 + (y'')^2\right]/2w^2\right\}\exp\left\{i\left[\varphi(y') - \varphi(y'')\right]\right\}& \\ \exp\left[-i\left(\frac{k^2}{2k_L} - \frac{(k')^2}{2k_L}\right)z\right] \quad .&\end{aligned} \tag{3.15}$$

If we assume a Gaussian autocorrelation length,

$$\langle\exp\{i[\varphi(y') - \varphi(y'')]\}\rangle = \exp\left[-(y'-y'')^2/2l^2\right] \quad , \tag{3.16}$$

we conclude

$$\langle |E_L|^2 \rangle = \frac{E_0^2}{2\pi w^2} \; \frac{w}{\left[w^2 + \left(\frac{l^2 + 2l^2 w^2}{l^4 w^2} \right) \frac{z^2}{k_L^2} \right]^{1/2}}$$
$$\exp \left(- \frac{y^2}{\left[w^2 + \left(\frac{l^4 + 2l^2 w^2}{l^4 w^2} \right) \frac{z^2}{k_L^2} \right]} \right) \quad . \tag{3.17}$$

In the limit where $l \gg w$, we see that Eq. (3.17) goes over to the standard result where the pulse spreads to $\sqrt{2}$ times its width over a length $z = k_L w^2$. In the limit where $w \gg l$, so that the beam is highly aberrated, the pulse spreads to $\sqrt{2}$ times its original length over a distance $z = k_L w l/\sqrt{2}$.

The phase aberrations rapidly translate into amplitude aberrations so that the intensity fluctuates as it spreads. To determine the size of these fluctuations, one must in principal calculate

$$\langle |E_L|^4 \rangle - \left(\langle |E_L| \rangle^2 \right)^2 \quad .$$

A detailed calculation of this quantity is rather messy, but intuitively we expect that $\langle |E_L|^4 \rangle \simeq (|E_L|^2)^2$ at a distance short compared to the aberration Fresnel length, $d_F = l^2 k_L$ and $\langle |E_L|^4 \rangle \simeq 2\left(|E_L|^2 \right)^2$ at distances long compared to d_F. Roughly speaking, we can consider $\mathbf{E}_L$ viewed as a function of y to vary on a length scale l. If $w = nl$, where n is some integer, then the original beam has n independent emitters, $\mathbf{E}_i$, $i = 1, n$. At a distance $z = Nl$, the number of emitters which contributes is roughly $N = z/l$ if $N < n$ or n otherwise. We now find

$$\mathbf{E}_L = \sum_{i=1}^{N} \mathbf{E}_i$$
$$\Rightarrow \; \langle |E_L(z)|^2 \rangle = \sum_{i=1}^{N} \langle |E_i(z)|^2 \rangle = N \langle |E_i(z)|^2 \rangle \quad , \tag{3.18}$$

where we assume that the expectation for each individual emitter is the same. Writing now

$$|E_L|^4 = \left(\sum_{i=1}^{N} \mathbf{E}_i\right) \cdot \left(\sum_{j=1}^{N} \mathbf{E}_j^*\right) \left(\sum_{k=1}^{N} \mathbf{E}_k\right) \cdot \left(\sum_{l=1}^{N} \mathbf{E}_l^*\right) \quad , \tag{3.19}$$

we conclude

$$\begin{aligned}\langle |E_L|^4 \rangle &= 2\left(\sum_{i=1}^{N} \mathbf{E}_i \cdot \mathbf{E}_i^*\right) \left(\sum_{k=1}^{N} \mathbf{E}_k \cdot \mathbf{E}_k^*\right) - \sum_{i=1}^{N} (\mathbf{E}_i \cdot \mathbf{E}_i^*)^2 \\ &= 2(N^2 - N)\big(\langle |E_i|^2 \rangle\big)^2 \simeq 2\langle |E_L|^2 \rangle^2\end{aligned} \tag{3.20}$$

when N is large. To pin down the connection with the Fresnel length, we note that when z is small

$$E_L(z) - E_L(0) = \frac{i}{2k_L} \left.\frac{\partial^2 E_L}{\partial y^2}\right|_{z=0} z \tag{3.21}$$

where the second derivative is evaluated at $z = 0$. Using Eq. (3.13), we now find

$$|E_L(z)|^2 = |E_L(0)|^2 - \frac{\varphi''(y)}{2k_L} z |E_L(0)|^2 \quad . \tag{3.22}$$

Noting that $|\varphi''(y)| \sim 1/l^2$, we conclude that the amplitude aberrations appear over a length d_F.

Once amplitude aberrations appear, they can have a deleterious affect on collinear beam amplification, particularly in the high gain regime. When $gE_0^2 d_F/4\pi w^2 \gg 1$ and $w \gg l$, then we are in the high gain, highly aberrated regime. We may ignore to lowest order the effect of diffraction on the Stokes beam amplification. In the central part of the beam where $y \ll w$, we may write

$$\begin{aligned}\frac{dE_S}{dz} &= \frac{gE_0^2}{4\pi w^2} \exp(-y^2/w^2) a(y) E_S \\ &\simeq \frac{gE_0^2}{4\pi w^2} a(y) E_S \quad ,\end{aligned} \tag{3.23}$$

where $a(y) = |E_L|^2/\langle |E_L|^2 \rangle$. The quantity $a(y)$ is Gaussian-distributed and $\langle a(y) \rangle = 1$. Specifically,

$$f(a) = \frac{\pi}{2} a \exp(-\pi a^2/4) \tag{3.24}$$

gives the probability distribution function of a. We then find

$$\frac{<|E_S|^2(z)>}{<|E_S|^2(0)>} \simeq \frac{gE_0^2 z}{2\pi w^2} \exp\left(\frac{g^2 E_0^4 z^2}{16\pi^3 w^4}\right) \qquad (III.25)$$

at large z. In effect, the amplitude aberrations in the pump lead to high amplitude spikes which in turn leads to differential growth in the Stokes and substantial spikiness. We have not carried out a calculation in which we determine the increase in the Stokes bandwidth, but it is clear that the Stokes can become more aberrated than the original pump, a case sometimes observed in practice.[6]

In the future, we intend to carry out a series of numerical studies using RAM2D1, aimed at verifying some of these theoretical results in the stationary regime.

III.C. Solitons and the Spectral Transformation

If we consider the usual transient equations, Eq. (II.A.1), in the limit of pulses very short compared to T_2 so that we may set $\Gamma = 0$, we obtain the following solitary wave solution

$$\begin{aligned} E_L &= a\,\mathrm{sech}(\alpha z - \beta t) \quad , \\ E_S &= \sqrt{\frac{k_S}{k_L}}\kappa_1 a \tanh(\alpha z - \beta t) \quad , \\ Q &= -\sqrt{\frac{k_S}{k_L}}\kappa_1 \frac{a^2}{\beta}\mathrm{sech}(\alpha z - \beta t) \quad , \end{aligned} \qquad (3.26)$$

where $\beta = \kappa_1 \kappa_2 a^2/\alpha$ which implies that the pulse is sub-luminous. This pulse has the remarkable property that $|E_S|$ tends to a non-zero value as $t \to \pm\infty$. Unfortunately, this property is not physical in the limit of short pulses. Nonetheless, it can be effectively true in situations where pulses are initially long compared to T_2, and the Stokes pulse undergoes a rapid phase flip at some point. It is in this context that solitons have been observed experimentally.[7,8] Indeed, there are theoretical considerations which indicate that dissipation plays an important role in the soliton's formation.[9,10] Virtually all theoretical work to date has focussed on the case where the original Stokes pulse has a 180° phase flip. It is of some interest to determine how close to 180° the phase flip must be before a soliton, or more precisely a soliton-like structure, will form. We have considered this issue and intended to report on it at the July '88 IQEC meeting in Tokyo, Japan. (The high cost of travel to Japan has prevented us from attending.) The summary for this meeting is included in Appendix D, and we do not repeat the details here. We conclude that if

$$E_S = K\Gamma_S t + iK_S \tag{3.27}$$

in the neighborhood of the phase flip, then a soliton will form if

$$\frac{\Gamma}{\Gamma_S}\,\frac{K_S}{K} < 1 \ . \tag{3.28}$$

Another issue of some importance is the possible generation of a series of solitons when a series of pump pulses is injected into a Raman cell. The evolution of a series of pulses has been considered by Reintjes, *et al.*[11] and it is of some interest to determine whether a series of solitons can emerge from these pulses. We have not considered this issue in any detail, but it is of some interest to point out that the transient equations do have periodic solutions when $\Gamma = 0$. Typical solutions are

$$\begin{aligned} E_L &= a\,\mathrm{cn}(\alpha z - \beta t|m) \ , \\ E_S &= \sqrt{\frac{k_S}{k_L}}\,a\,\mathrm{sn}(\alpha z - \beta t|m) \ , \\ Q &= -\sqrt{\frac{k_S}{k_L}}\,\frac{a^2}{m\beta}\,\mathrm{dn}(\alpha z - \beta t|m) \ , \end{aligned} \tag{3.29}$$

where $\alpha\beta = \kappa_1\kappa_2 a^2/m$. Whether this solution is realizable in practice will be determined in future investigations.

We now turn to a discussion of the spectral transform method which applies in the limit of short, transient pulses. There are theoretical reasons to suspect that solitons in these systems are always transient. On physical grounds, we might anticipate this result as the quantity

$$K = |E_L|^2 + \frac{k_L}{k_S}|E_S|^2 \tag{3.30}$$

is constant at every t-point while solitons are sub-luminous. Hence, we expect them to disappear at the back end of the pulse. We shall see that the spectral transform method has peculiarities in our case which result in solutions of very different character from those which are normally found when spectral methods can be used.

We first make a change of variables so that our notation follows that normally used in this field.[9,12,13] We let $A_1 = E_L$, $A_2 = (k_L/k_S)^{1/L}E_S$, $X = i(k_L/k_S)^{1/2}Q$, $\epsilon = \Gamma/\kappa_1$, $\tau = \kappa_1 t$, and $\chi = \kappa_2 z$, yielding

$$\begin{aligned} \frac{\partial A_1}{\partial \chi} &= -XA_2 \ , \\ \frac{\partial A_L}{\partial \chi} &= XA_1 \ , \\ \frac{\partial X}{\partial \tau} + \epsilon X &= A_1 A_2^* \ . \end{aligned} \tag{3.31}$$

We apply the spectral transform approach in the limit where we may set $\epsilon = 0$. Following Kaup,[9] we first consider two new quantities u_1 and u_2 which satisfy the equations

$$\begin{aligned} \frac{\partial u_1}{\partial \chi} - \frac{i}{\varsigma}u_1 &= \chi u_2 \ , \\ \frac{\partial u_2}{\partial X} + \frac{i}{\varsigma}u_2 &= -X^* u_2 \ , \end{aligned} \tag{3.32}$$

and

$$\begin{aligned} \frac{\partial u_1}{\partial \tau} + i\varsigma S_3 u_1 &= \varsigma S_+ u_2 \ , \\ \frac{\partial u_2}{\partial \tau} - i\varsigma S_3 u_2 &= -\varsigma S_- u_1 \ , \end{aligned} \tag{3.33}$$

where

$$\begin{aligned} S_3 &= \frac{1}{4}(A_1 A_1^* - A_2 A_2^*) \ , \\ S_+ &= \frac{i}{2}A_2^* A_1 \ , \\ S_- &= S_+^* \ . \end{aligned} \tag{3.34}$$

Equations (3.32) and (3.33) are *compatible*, *i.e.*, their cross-derivatives are equal, only if Eq. (3.31) holds with $\varepsilon = 0$. At this point, we define the quantities

$$A = \frac{1}{4}(A_1 A_1^* + A_2 A_2^*) \quad , \tag{3.35}$$

the angles β and θ through the relations

$$\begin{aligned} S_3 &= A\cos\beta \quad , \\ S_+ &= Ae^{i\theta}\sin\beta \quad , \end{aligned} \tag{3.36}$$

and the angle γ through the compatibility relations

$$\begin{aligned} \frac{\partial\gamma}{\partial\tau} &= \cos\beta\frac{\partial\theta}{\partial\tau} \quad , \\ \frac{\partial\gamma}{\partial\chi} &= \frac{2}{\sin\beta}(X_1\cos\theta - X_2\sin\theta) \quad , \end{aligned} \tag{3.37}$$

where we have decomposed

$$X = X_1 - iX_2 \quad . \tag{3.38}$$

We define as well the matrices,

$$\begin{aligned} \Gamma &= I\cos(\gamma/2) + i\sigma_3\sin(\gamma/2) \quad , \\ B &= I\cos(\beta/2) + i\sigma_1\sin(\beta/2) \quad , \\ \Theta &= I\cos(\theta/2) + i\sigma_3\sin(\theta/2) \quad , \end{aligned} \tag{3.39}$$

where σ_1, σ_2, and σ_3 are the Pauli matrices. Finally, we let

$$V = \begin{pmatrix} u_1 & \hat{u}_1 \\ u_2 & \hat{u}_2 \end{pmatrix} \quad , \tag{3.40}$$

where (u_1, u_2) and $(\hat{u}_1, \hat{u}_2)$ are two independent solutions of Eq. (3.33). Making the transformation

$$V = \Gamma B \Theta^{-1} U \quad , \tag{3.33}$$

we have verified after substantial algebra that V satisfies the equation

$$\left(I\frac{\partial}{\partial T} + i\varsigma\sigma_3\right)V = \begin{pmatrix} 0 & q \\ -q^* & 0 \end{pmatrix} V \quad , \tag{3.42}$$

where

$$T = \int_{-\infty}^{\tau} A\, d\tau' \quad ,$$
$$q = \frac{i}{2\cos\beta}\frac{\partial}{\partial T}[e^{i\gamma}\sin\beta] \quad . \tag{3.43}$$

Equation (3.42) has the standard form of the AKNS systems.[14] We impose the boundary condition $V(\tau \to -\infty) = \sigma_3$ and let $V(\tau \to +\infty) = YS^t\sigma_3$, where

$$Y = \begin{pmatrix} e^{-i\varsigma T_\infty} & 0 \\ 0 & e^{i\varsigma T_\infty} \end{pmatrix} , \qquad S = \begin{pmatrix} a & b \\ -\bar{b} & \bar{a} \end{pmatrix} , \tag{3.44}$$

a, $\bar{a}$, b, $\bar{b}$ are the usual scattering coefficients, and $T_\infty = T(\tau = +\infty)$. We now find

$$\frac{\partial V}{\partial \chi} = \frac{i}{\varsigma}\Gamma B\sigma_3 B^{-1}\Gamma^{-1}V - \frac{i}{\varsigma}V\sigma_3\Gamma_0 B_0\sigma_3 B_0^{-1}\Gamma_0^{-1}\sigma_3 \quad , \tag{3.45}$$

where the subscripted matrices, B_0 and Γ_0, are B and Γ in the limit $\tau \longrightarrow -\infty$. Once the evolution of V is known in the limit $\tau = \infty$, we can determine $a(\chi)$, $\bar{a}(\chi)$, $b(\chi)$, and $\bar{b}(\chi)$. Where q is compact, *i.e.* T_∞ is finite, a, $\bar{a}$, b, and $\bar{b}$ are all analytic as a function of ς. We now define

$$G(x) = \frac{1}{2\pi}\int_C \frac{\bar{b}}{a}\, e^{-i\varsigma x}\, d\varsigma \quad ,$$
$$\bar{G}(x) = \frac{1}{2\pi}\int_{\bar{C}} \frac{b}{\bar{a}}\, e^{i\varsigma x}\, d\varsigma \quad , \tag{3.46}$$

where the contour C goes over all the zeroes of a and $\bar{C}$ goes under all the zeroes of $\bar{a}$. Solving the linear equations

$$\bar{L}(x,y) + \begin{pmatrix}1\\0\end{pmatrix}G(x+y) - \int_{-\infty}^{x} L(x,s)G(s+y)\, ds = 0 \quad ,$$
$$L(x,y) + \begin{pmatrix}0\\1\end{pmatrix}\bar{G}(x+y) + \int_{-\infty}^{x} \bar{L}(x,s)\bar{G}(s+y)\, ds = 0 \quad , \tag{3.47}$$

we can find $q(x)$ using the relation

$$q(x) = 2\bar{L}_1(x,x) \quad . \tag{3.48}$$

In standard AKNS systems, the evolution of the scattering is quite simple, while the zeroes of a and $\bar{a}$ are fixed and correspond to solitons.[14] In our problem, that is no longer the case as $X \neq 0$ in general in the limit $\tau \to \infty$, and, hence, the evolution of the spectral data cannot be easily determined. Fortunately, Kaup[12] has shown that solving the equation

$$\frac{\partial V}{\partial \chi} = -\frac{i}{\varsigma} V \sigma_3 \Gamma_0 B_0 \sigma_3 B_0^{-1} \Gamma_0 \sigma_3 \quad , \tag{3.49}$$

will still yield the correct answer for q when the previously outlined procedure is followed. However, the spectral data obtained in this way is not true spectral data. The zeroes of a and $\bar{a}$ are not fixed and no longer correspond to solitons. To illustrate this point, we consider a simple example already studied by Duncan, *et al.*[5] We suppose that the Stokes is initially a multiple of the pump. We then find that

$$q(\chi = 0) = 0 \quad . \tag{3.50}$$

From Eq. (3.49), we find

$$\begin{aligned} a_\chi &= -\frac{i}{\varsigma}[a \cos\beta_0 - i\bar{b}\sin\beta_0] \quad , \\ \bar{b}_\chi &= -\frac{i}{\varsigma}[ia \sin\beta_0 - \bar{b}\cos\beta_0] \quad , \\ \bar{a}_\chi &= \frac{i}{\varsigma}[\bar{a}\cos\beta_0 + ib\sin\beta_0] \quad , \\ b_\chi &= -\frac{i}{\varsigma}[i\bar{a}\sin\beta_0 + b\cos\beta_0] \quad . \end{aligned} \tag{3.51}$$

Using Eq. (3.50), it now follows that

$$G(2T) = \frac{1}{2\pi}\int_C \frac{i\sin\beta_0[e^{-i\chi/\varsigma} - e^{i\chi/\varsigma}]}{(1-\cos\beta_0)e^{i\chi/\varsigma} + (1+\cos\beta_0)e^{-i\chi/\varsigma}} e^{2i\varsigma T} d\varsigma \quad . \tag{3.52}$$

When $\beta_0 \simeq 0$, the zeroes of the integrand lie in the upper half plane, there are an infinite number of them clustering about the essential simularity at $\varsigma = 0$, and they explode outward as χ increases. We have yet to make a complete evaluation of Eq. (3.52), not to mention a determination of $L(x,y)$ and $\bar{L}(x,y)$. We may consider this problem in the future.

It is possible to show however that when χ is small, the usual linear result is reproduced. Expanding the integrand of Eq. (3.52) as a power series in β_0, assuming that it is small, we find

$$G(2T) \simeq \frac{\beta_0}{4\pi i} \int_+ (1 - e^{2i\chi/\varsigma}) e^{-2i\varsigma T} \, d\varsigma \quad , \tag{3.53}$$

where the contour + is a small, positive circle around $\varsigma = 0$. Recalling the relation

$$\exp\left[\frac{1}{2} y(t + 1/t)\right] = \sum_{k=-\infty}^{\infty} t^k I_k(y) \quad , \tag{3.54}$$

we find

$$G(2T) \simeq -\frac{i\beta_0}{2} \left(\frac{\chi}{T}\right)^{1/2} I_1\left[4(\chi T)^{1/2}\right] \quad . \tag{3.55}$$

Writing now

$$i\frac{d\beta}{dT} = q(T) = 2\bar{L}_1(T,T) \simeq -2G(2T) \quad , \tag{3.56}$$

we finally conclude

$$\beta(T) = \beta_0 I_0\left[4(\chi T)^{1/2}\right] \quad , \tag{3.57}$$

a result which had earlier been obtained by Duncan *et al.*[5] using more elementary methods.

REFERENCES

1 See, *e.g.*, T. R. Taha and M. Ablowitz, J. Comp. Phys **55**, 203 (1984).

2 See, *e.g.*, C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*, (Prentice-Hall, Englewood Cliffs, NJ, 1971).

3 C. S. Wang, Phys. Rev **182**, 482 (1969).

4 R. L. Carmen, F. Shimuzu, C. S. Wang, and N. Bloembergen, Phys. Rev A **2**, 60 (1970).

5 M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, JOSA B **5**, 37 (1988).

6 Dr. J. Reintjes, private communication.

7 K. J. Druhl, R. G. Wenzel, and J. L. Carlsten, Phys. Rev. Lett. **51**, 1171 (1983).

8 R. G. Wenzel, J. L. Carlsten, and K. J. Druhl, J. Stat. Phys. **39**, 621 (1985).

9 D. J. Kaup, Physica **19D**, 125 (1986).

10 K. J. Druhl, J. L. Carlsten, and R. G. Wenzel, J. Stat. Phys. **39**, 615 (1985).

11 J. Reintjes, M. D. Duncan, G. Calame, L. L. Tankersley, and R. Mahon, Opt. News **39**, 101 (1987).

12 D. Kaup, Physica **6D**, 143 (1983).

13 H. Steudel, Physica **6D**, 155 (1983).

14 See, *e.g.*, M.J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering Transform* (SIAM, Philadelphia, 1981).

APPENDIX A

Code Listings

Typical namelists and output are included in the manual (Appendix B).

RAM2D1 (version C)

```
      PROGRAM RAM2D1C
C
C THIS IS VERSION C OF THE TRANSIENT RAMAN AMPLIFIER CODE
C R A M 2 D 1. THIS VERSION IS ADAPTED TO RUN ON THE CENTRAL
C COMPUTING FACILITY AT THE NAVAL RESEARCH LABORATORY.
C
C RAM2D1 WAS WRITTEN BY CURTIS R. MENYUK 11/86 TO SOLVE THE
C COUPLED RAMAN EQUATIONS WITH BOTH TRANSIENT AND DIFFRACTIVE
C PHENOMENA ACCOUNTED FOR.  THE EQUATIONS ARE ADVANCED IN THE
C Z-DIRECTION.  THE BEHAVIOR IN THE TIME "DIRECTION" AND THE
C TRANSVERSE (Y) DIRECTION ARE DETERMINED AT EACH Z-STEP.  HENCE,
C THIS CODE IS 2+1D.  THE TWO QUANTITIES THAT ARE ADVANCED AT EACH
C STEP ARE THE PUMP AND THE STOKES AMPLITUDES.  IN ADDITION, THE
C MATERIAL EXCITATION MUST BE DETERMINED AT EACH STEP THROUGH A
C CONSTRAINT EQUATION.
C
C THE SYSTEM OF EQUATIONS IS SOLVED USING A SEMI-SPECTRAL APPROACH.
C THE Y-DERIVATIVES OF THE DYNAMICAL EQUATIONS (THE EQUATIONS
C GOVERNING THE PUMP AND STOKES WAVES) ARE DETERMINED IN KY-SPACE,
C AND THE NONLINEAR TERMS ARE DETERMINED IN Y-SPACE.  THERE ARE NO
C T-DERIVATIVES IN THE DYNAMICAL EQUATIONS AND HENCE NO NEED TO USE
C OMEGA SPACE.  THE CONSTRAINT EQUATION (THE EQUATION GOVERNING
C THE MATERIAL EXCITATION) DOES CONTAIN A T-DERIVATIVE BUT NO
C Y-DERIVATIVE.
C
C THE CONSTRAINT EQUATION IS SOLVED FOR Q(EL,ES), SUBJECT TO THE
C APPROPRIATE BOUNDARY CONDITION (THE MATERIAL EXCITATION IS ZERO
C WHEN T GOES TO MINUS INFINITY).  IT IS SOLVED FOR IN ONE OF THREE
C WAYS, DEPENDING ON THE PARAMETER REGIME:  1) WHEN NT IS EIGHT OR
C LESS, A SET OF 1-D STATIONARY CASES IS RUN.  2) WHEN MAX(ABS(T/TTWO))
C < 10.0 AND NT > 8, A RUNNING SUM IS PERFORMED.  IN THIS APPROACH, IT
C IS NOT NECESSARY THAT Q=0 AT TMAX.  3) WHEN MAX(ABS(T/TTWO)) > 10.0
C AND NT > 8, A FOURIER TRANSFORM APPROACH IS USED.
C
C THE DYNAMICAL EQUATIONS ARE ADVANCED IN Z USING A MIDPOINT EULER
C METHOD WITH ONE SPECIAL MODIFICATION - THE LINEAR PORTION OF EACH
C EQUATION IS ADVANCED IN SUCH A WAY THAT IT IS SOLVED EXACTLY TO
C WITHIN ROUNDOFF.  IN THIS VERSION, THE STEP SIZE IS FIXED.
C
C ------------------------------------------------------------------
C TIME IS DIMENSIONED IN PICOSECONDS; DISTANCE IS DIMENSIONED IN
C CENTIMETERS; AND POWER IS DIMENSIONED IN GIGAWATTS.  ALL OTHER
C QUANTITIES ARE CORRESPONDINGLY DIMENSIONED.
C ------------------------------------------------------------------
C
C ******************************************************************
C MODIFICATION 4/87:
C THIS PROGRAM ASSUMES THAT WHEN NY IS EIGHT OR LESS, A SET OF 1-D
C TRANSIENT CASES WITH NO Y-VARIATION ARE BEING RUN.
C NOTE:  ONE MUST SET ICOND=3 IN THIS CASE FOR THE PROGRAM TO
C        INITIATE PROPERLY
C ******************************************************************
C MODIFICATION 5/87:
C THIS PROGRAM ASSUMES THAT WHEN NT IS EIGHT OR LESS, A SET OF
C STATIONARY CASES WITH NO T-VARIATION ARE BEING RUN.  IN THIS
C CASE CFFT2 IS CALLED TO CARRY OUT THE FOURIER TRANSFORMS
C SERIALLY, RATHER THAN CARRYING THEM OUT IN PARALLEL AS IN THE
C 2-D CASE.
C NOTE:  ONE MUST SET ICOND=4 IN THIS CASE FOR THE PROGRAM TO
C        INITIATE PROPERLY
C ******************************************************************
C MODIFICATION 9/87:
C THE DATA OUTPUT FILE NAME WAS CHANGED FROM 'FRAM' TO THE FOLLOWING:
```

```
C     THE DATA FILE NAME'S FIRST CHARACTER (F) STANDS FOR THE OLD DATA
C     FILE NAME 'FRAM'. THE SECOND CHARACTER INDICATES THE T-DIMENSION,
C     THE THIRD THE Y-DIMENSION. THE DIMENSIONS ARE REPRESENTED BY THEIR
C     NUMBER (1-8) IF LESS THAN 9. IF GREATER THAN 8 THE DIMENSIONS ARE
C     ASSUMED TO BE INTEGRAL POWERS OF 2. THE N-TH POWER OF 2 IS
C     REPRESENTED BY THE N-TH CHARACTER OF RLFBET. THE FOURTH THROUGH
C     NINETH CHARACTER IN THE FILE NAME ENCODES THE MONTH, DAY, AND YEAR
C     THE PROGRAM WAS STARTED. A THENTH THROUGH TWELFTH CHARACTER IS
C     APPENDED, NUMBERING THE PARTIAL DATA FILES THAT ARE GENERATED
C     WHEN THE PROGRAM RUNS TWO-DIMENSIONALLY (MAXIMALLY 999 NEW FILES).
C     *********************************************************************
C
C                                     --VARIABLES--
C
C         NY = NUMBER OF Y POINTS (MUST BE A POWER OF 2)
C         NT = NUMBER OF T POINTS (MUST BE A POWER OF 2)
C         NP = MAXIMUM NUMBER OF PUMP BEAMS
C         NPUMP = ACTUAL NUMBER OF PUMPS
C         YM = DELIMITING Y-VALUES (CM)
C         TM = DELIMITING T-VALUES (PS)
C         ZINT = BEAM INTERSECTION POINT (CM)
C         RKP = PUMP WAVENUMBER (CM**-1)
C         RKS = STOKES WAVENUMBER (CM**-1)
C         YOFF = Y-OFFSETS OF THE PUMP BEAMS (CM)
C         TOFF = T-OFFSETS OF THE PUMP BEAMS (PS)
C         YWIDTH = Y-WIDTHS OF THE PUMP BEAMS (CM)
C         TWIDTH = T-WIDTHS OF THE PUMP BEAMS (PS)
C         YOST = Y-OFFSET OF THE STOKES BEAM (CM)
C         TOST = T-OFFSET OF THE STOKES BEAM (PS)
C         YWST = Y-WIDTH OF THE STOKES BEAM (CM)
C         TWST = T-WIDTH OF THE STOKES BEAM (PS)
C         RINT = INTENSITY OF THE PUMP BEAMS (GW/CM**2)
C         RIST = INTENSITY OF THE STOKES BEAM (GW/CM**2)
C         RAMP = AMPLITUDE OF THE PUMP BEAMS [SQRT(PS*GW/CM**3)]
C         RIST = AMPLITUDE OF THE STOKES BEAM [SQRT(PS*GW/CM**3)]
C         RAMASM = AMPLITUDE OF THE STOKES ASSYMETRY
C         RALASM = STRETCH OF THE STOKES ASSYMETRY
C         NHYP = EXPONENT OF THE HYPERGAUSSIAN DISTRIBUTION IN THE
C                Y-DIRECTION (MUST BE AN EVEN INTEGER)
C         PHL = FACTOR MULTIPLYING THE INITIAL PUMP CHIRP
C         PHST = FACTOR MULTIPLYING THE INITIAL STOKES CHIRP
C         TOC = CHIRP PULSE TIME OFFSET (PS)
C         TWC = CHIRP PULSE T-WIDTH [SET TO TWIDTH(1)] (PS)
C         YWC = CHIRP PULSE Y-WIDTH [SET TO YWIDTH(1)] (PS)
C         ICOND = TYPE OF INITIAL PUMP AND STOKES PROFILES
C               = 1:  DOUBLE-SECH PROFILE
C               = 2:  SECH**2-HYPERGAUSSIAN PROFILE
C               = 3:  1-D TRANSIENT CASES (NO Y-VARIATION)
C               = 4:  STATIONARY CASE (TO T-VARIATION)
C         ZSTEP = STEP SIZE (CM)
C         ZH = ZSTEP/2 (CM)
C         ZFINAL = FINAL Z-VALUE (CM)
C         ZKEEP = Z-VALUE INCREMENT BETWEEN POINTS WHERE DATA IS
C                 STORED (CM)
C         NMAX = MAXIMUM NUMBER OF ALLOWED STEPS IN Z (MUST BE LESS THAN
C                OR EQUAL TO NST)
C         TTWO = DAMPING TIME OF THE MATERIAL EXCITATION (PS)
C         GAIN = RAMAN GAIN FACTOR ASSUMING NO PUMP DEPLETION (CM/GW)
C         RKAP1 = KAPPA-1:  NONLINEAR COEFFICIENT IN THE MATERIAL
C                 EXCITATION EQUATION [SQRT(CM**3/GW*PS)]
C         RKAP2 = KAPPA-2:  NONLINEAR COEFFICIENT IN THE STOKES
C                 EQUATION [SQRT(CM**3/GW*PS)/CM*PS]
C         SPEED = SPEED OF LIGHT IN VACUUM (USED IN THIS CODE TO
C            APPROXIMATE THE SPEED OF LIGHT IN THE MATERIAL)  (CM/PS)
C      EL = PUMP ARRAY (NT*NY)
C      ES = STOKES ARRAY (NT*NY)
C      Q = MATERIAL EXCITATION ARRAY (NT*NY)
C      AEL,AES,AQ = SORRESPONDING FOURIER ARRAYS
C      AW = STORAGE ARRAY FOR THE MIDPOINT EULER METHOD (NT*NY*4)
C      CW = WORKING ARRAY FOR THE Y-DIRECTION FFT (5*NY/2)
C          (LENGTH MODIFICATION MADE 5/87 TO ALLOW CFFT2 TO RUN)
C      CWQ = WORKING ARRAY FOR THE T-DIRECTION FFT (NT)
C          USED WITH METHOD 2 OF CONSTRAINT EQ. SOLN.
C      WQ1,WQ2 = WORKING ARRAYS FOR CONSTRAINT EQ. SOLN. WITH
C                METHOD 1 (NT).  THEY ARE EQUIVALENCED WITH CWQ
C                TO CONSERVE SPACE
C      COMVEC = INVERSE KERNEL FOR THE MATERIAL EXCITATION EQUATION
C               WHEN METHOD 2 IS USED (NT)
C      CYVEC = KERNEL FOR THE SECOND ORDER Y-DERIVATIVE OPERATOR IN
C              THE DYNAMICAL EQUATIONS.  THE FINITE LENGTH IS ACCOUNTED
C              FOR SO THAT IN THE LINEAR LIMIT THE PROPAGATOR IS EXACT.
C              TWO VECTORS ARE NEEDED (NY*2)
C      NWRT = NUMBER OF RECORD GROUPS IN UNIT 4
C ------------------------------------------------------------------
C
C         --VARIABLES, BOTH ALTERED AND NEW, 1-D TRANSIENT CASE--
C
C      NY = ACTUAL NUMBER OF CASES RUN
C      ITYPE = TYPE OF INITIAL PROFILE
C            = 1:  SECH PROFILE
C            = 2:  RECTANGULAR PROFILE
C            = 3:  LORENTZIAN PROFILE
C            = 4:  EXPONENTIAL PROFILE
C      RTYPE = POWER TO WHICH PROFILE IS TAKEN (ITYPE = 1 & 3)
C            = POWER TO WHICH EXPONENT IS TAKEN (ITYPE = 4)
C ------------------------------------------------------------------
C
C         --VARIABLES, BOTH ALTERED AND NEW, STATIONARY CASE--
C
C      NT = ACTUAL NUMBER OF CASES RUN
C      AW1,AW2 = WORKING ARRAYS USED BY CFFT2 (NY)
C      RABAMP = FRACTIONAL CONTRIBUTION OF THE AMPLITUDE ABERRATIONS
C      RDSLIM = NUMBER OF TIMES DISPERSION LIMITED THE PUMP BEAMS ARE
C               DUE TO ABERRATIONS
C               [SET WITH RESPECT TO YWIDTH(1)]
C ------------------------------------------------------------------
C
      PARAMETER(NT=256,NY=256,NTHP=1+NT/2,NP=10,NPM2=NP-2,NST=4000,
     1 NS=5*NY/2)
C
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
     1 AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),CYVEC(NY,2),
     2 COMVEC(NT),CWQ(NT),WQ1(NT),WQ2(NT),YWIDTH(NP),TWIDTH(NP),
     3 YOFF(NP),TOFF(NP),RAMP(NP),RINT(NP),PHL(NP),ITYPE(8),RTYPE(8),
     4 RABAMP(8),RDSLIM(8),SFE(NST),SQ(NST),SSTEP(NST),YM(2),TM(2)
      CHARACTER*1 D1
      CHARACTER*2 D1A
      CHARACTER*2 D2
      CHARACTER*2 D3
      CHARACTER*7 DTFL0
      CHARACTER*7 DTFL1
      CHARACTER*6 DTFL1D
      CHARACTER*7 DTFL2D
      CHARACTER*8 FDATE
      CHARACTER*9 FRAM
      CHARACTER*6 FRM
      CHARACTER*1 ISTP1
      CHARACTER*1 ISTP2
      CHARACTER*1 ISTP3
      CHARACTER*10 NUMRAL
      CHARACTER*9 PDN1D
      CHARACTER*12 PDN2D
      CHARACTER*12 PDN0
      CHARACTER*12 PDN1
      CHARACTER*26 RLFBET
      CHARACTER*1 TDIM
      CHARACTER*1 YDIM
      EQUIVALENCE (CWQ,WQ1),(CWQ(NTHP),WQ2)
      NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
     2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
      COMMON/VINIT/NPUMP,YM,TM,ZINT,YOFF,TOFF,YWIDTH,TWIDTH,YOST,TOST,
     1 YWST,TWST,RAMP,RAST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,ITYPE,RTYPE,
     2 RABAMP,RDSLIM
      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
      COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT
C
      DATA PI/3.14159265358979/,SPEED/0.0299779/
      DATA YM/-0.3,0.3/,TM/-100.0,100.0/,NPUMP/2/,GAIN/3.0/,
     1 RINT/NP*0.55/,RIST/0.003/,TTWO/633.0/,YOFF/0.14,-0.14,NPM2*0.0/,
     2 TOFF/NP*0.0/,YWIDTH/NP*0.10/,TWIDTH/NP*40.0/,YOST/0.0/,
     3 TOST/-40.0/,YWST/0.10/,TWST/40.0/,RAMASM/1.5/,RALASM/5.0/,
     4 NHYP/8/,PHL/NP*0.0/,TOC/5.0/,PHST/0.0/,ICOND/2/,ITYPE/8*1/,
     5 RTYPE/8*2.0/,RABAMP/8*0.0/,RDSLIM/8*1.0/,ZINT/20.0/,
     6 RKP/1.180E+5/,RKS/0.91893E+5/,ZSTEP/0.05/,ZFINAL/50.0/,
     7 ZKEEP/1.0/,NMAX/4000/
C
      CALL ASSIGN(IRRE,'DN'L,'ERRM'L,'A'L,'FT59'L)
      RLFBET='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
C             12345678901234567890123456
      NUMRAL='0123456789'
      IF (NT.GT.8) THEN
         ITDIM=NINT(ALOG(FLOAT(NT))/ALOG(2.0))
         TDIM=RLFBET (ITDIM:ITDIM)
      ELSE
         TDIM=NUMRAL (NT+1:NT+1)
      ENDIF
      IF (NY.GT.8) THEN
         IYDIM=NINT(ALOG(FLOAT(NY))/ALOG(2.0))
         YDIM=RLFBET (IYDIM:IYDIM)
      ELSE
         YDIM=NUMRAL (NY+1:NY+1)
      ENDIF
      CALL DATE(NDATE)
      WRITE (FDATE,'(A8)') NDATE
      D1=FDATE (2:2)
      D1A=FDATE (1:2)
      IF (D1A.EQ.'10') D1='A'
      IF (D1A.EQ.'11') D1='B'
      IF (D1A.EQ.'12') D1='C'
      D2=FDATE (4:5)
      D3=FDATE (7:8)
      FRM='F'//TDIM//YDIM//D1//D2
      FRAM='F'//TDIM//YDIM//D1A//D2//D3
      IF (NT.GT.8.AND.NY.GT.8) THEN
         ISTP1=NUMRAL(1:1)
         ISTP2=NUMRAL(2:2)
         DTFL0=FRM//ISTP1
```

```
            DTFL1=FRM//ISTP2
            CALL ASSIGN(IRRE,'DN'L,DTFL0,'A'L,'FT02'L)
            WRITE (59,*) 'ASSIGN FT02= ',DTFL0
            CALL ASSIGN(IRRE,'DN'L,DTFL1,'A'L,'FT03'L)
            WRITE (59,*) 'ASSIGN FT03= ',DTFL1
            PDN0=FRAM//ISTP1//ISTP1//ISTP1
            PDN1=FRAM//ISTP1//ISTP1//ISTP2
            IZNO=1
         ELSE
            DTFL1D=FRM
            CALL ASSIGN(IRRE,'DN'L,DTFL1D,'A'L,'FT04'L)
            WRITE (59,*) 'ASSIGN FT04= ',DTFL1D
            PDN1D=FRAM
         ENDIF
         CALL ASSIGN(IRRE,'DN'L,'NRAM'L,'A'L,'FT01'L)
         READ (1,NAML)
         CALL SECOND(STOT1)
C
C - SET KAPPA-FACTORS
         RKAP1=SQRT(GAIN/(RKS*(RKP-RKS)*TTWO))/(8.0*PI)
         RKAP2=4.0*PI*RKS*(RKP-RK[illegible]*SPEED*RKAP1
C
C - SET PUMP AND STOKES AMPLITUDES
         R1=8.0*PI/SPEED
         NAMP=NPUMP
         IF (NY.LE.8) NAMP=NY
         DO 5 I1=1,NAMP
 5       RAMP(I1)=SQRT(R1*RINT(I1))
         RAST=SQRT(R1*RIST)
C
C - MISCELLANEOUS INITIALIZATIONS, INCLUDING THE WORKING ARRAY FOR THE
C   Y-DIRECTION FFT
         ZFIN=ZFINAL-1.0E-06
         ZKP=ZKEEP-1.0E-06
         N999=AMOD(ZFINAL,ZKEEP)
         IF (N999.GE.998) THEN
            WRITE (59,*) 'DATA FILES IN EXCESS OF 999'
            CALL EXIT(1)
         ENDIF
         IF (NY.GT.8) CALL CFOUR2(EL,CW,NY,NT,1,0,AW1,AW2)
         ZVAL=0.0
         ZH=0.5*ZSTEP
C
C -- DETERMINE Y-SECOND-ORDER-DERIVATIVE KERNEL
C
         IF (NY.GT.8) THEN
            YFAC=2.0*PI/(YM(2)-YM(1))
            DO 8 I2=1,NY/2
 8          CYVEC(I2,2)=-0.5*(0.0,1.0)*ZH*((I2-1)*YFAC)**2
            DO 9 I2=1+NY/2,NY
 9          CYVEC(I2,2)=-0.5*(0.0,1.0)*ZH*((-NY+I2-1)*YFAC)**2
            DO 10 I2=1,NY
            CYVEC(I2,1)=CEXP(CYVEC(I2,2)/RKP)
            CYVEC(I2,2)=CEXP(CYVEC(I2,2)/RKS)
 10         CONTINUE
         ELSE
            YFAC=1
            DO 12 I2=1,NY
            CYVEC(I2,1)=1.0
            CYVEC(I2,2)=1.0
 12         CONTINUE
         ENDIF
C
```

```
C --- SET TRAT AND THE WORKING ARRAYS FOR DETQ
C
      IF (NT.GT.8) TRAT=AMAX1(-TM(1)/TTWO,TM(2)/TTWO)
C
C - IF METHOD 2, SET WQ1 AND WQ2
      IF (TRAT.LE.10.0.AND.NT.GT.8) THEN
         RDT=(TM(2)-TM(1))/NT
         DO 15 I3=1,NT
         TVAL=TM(1)+RDT*(I3-1)
         WQ1(I3)=EXP'TVAL/TTWO)
         WQ2(I3)=1.0/WQ1(I3)
 15      CONTINUE
C
C - IF METHOD 3, SET CWQ AND COMVEC
      ELSEIF (TRAT.GT.10.AND.NT.GT.8) THEN
         CALL CFOUR2(Q,CWQ,NT,NY,1,0,AW1,AW2)
         R1=1.0/TTWO
         R2=2.0*PI/(TM(2)-TM(1))
         R3=NT
         DO 17 I3=1,NT/2
 17      COMVEC(I3)=-(0.0,1.0)*RKAP1/((R1-(0.0,1.0)*(I3-1)*R2)*R3)
         DO 18 I3=NTHP,NT
 18      COMVEC(I3)=-(0.0,1.0)*RKAP1/((R1-(0.0,1.0)*(-NT+I3-1)*R2)*R3)
      ENDIF
C
C --- RECORD INITIAL DATA
C
      IF (NT.GT.8.AND.NY.GT.8) THEN
         WRITE (2) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1   YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
     2   TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
     3   TTWO,GAIN
         WRITE (3) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1   YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
     2   TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
     3   TTWO,GAIN
      ELSE
         WRITE (4) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1   YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
     2   TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,
     3   TTWO,GAIN
      ENDIF
      NWRT=1
C
C - DETERMINE CPU TIME FOR INIT
      CALL SECOND(SINIT1)
      CALL INIT(ICOND)
      CALL SECOND(SINIT2)
      SINIT=SINIT2-SINIT1
C
C - RECORD INITIAL COORDINATE DATA AND FOURIER DATA:  NOTE AQ=Q=0.0
      IF (NY.GT.8) THEN
         CALL SHFT(EL,NY,NT)
         CALL SHFT(ES,NY,NT)
         IF (NT.GT.8) THEN
            WRITE (2) ZVAL,EL
            WRITE (2) ZVAL,ES
            WRITE (2) ZVAL,Q
            WRITE (3) ZVAL,EL
            WRITE (3) ZVAL,ES
            WRITE (3) ZVAL,Q
            CLOSE (2)
            CALL SAVE(IRRE,'DN'L,DTFL0,'PDN'L,PDN0,
     1                'RESIDE'L,'OFFLINE'L)
            CALL RELEASE(IRRE,'DN'L,DTFL0)
         ELSE
            WRITE (4) ZVAL,EL
            WRITE (4) ZVAL,ES
            WRITE (4) ZVAL,Q
         ENDIF
         CALL SHFT(EL,NY,NT)
         CALL SHFT(ES,NY,NT)
      ELSE
         WRITE (4) ZVAL,EL
         WRITE (4) ZVAL,ES
         WRITE (4) ZVAL,Q
      ENDIF
      NWRT=NWRT+3
C
C -- DETERMINE INITIAL FOURIER DATA
C
      DO 20 I2=1,NY
      DO 20 I3=1,NT
      AEL(I3,I2)=EL(I3,I2)
      AES(I3,I2)=ES(I3,I2)
      AQ(I3,I2)=Q(I3,I2)
 20   CONTINUE
      IF (NY.GT.8) THEN
         CALL CFOUR2(AEL,CW,NY,NT,-1,1,AW1,AW2)
         CALL CFOUR2(AES,CW,NY,NT,-1,1,AW1,AW2)
         R1=1.0/(YFAC*NY)
         DO 30 I2=1,NY
         DO 30 I3=1,NT
         AEL(I3,I2)=R1*AEL(I3,I2)
         AES(I3,I2)=R1*AES(I3,I2)
 30      CONTINUE
C
C - RECORD INITIAL FOURIER DATA:  NOTE AQ=0.0
         CALL SHFT(AEL,NY,NT)
         CALL SHFT(AES,NY,NT)
         IF (NT.GT.8) THEN
            WRITE (2) ZVAL,AEL
            WRITE (2) ZVAL,AES
            WRITE (2) ZVAL,AQ
            WRITE (3) ZVAL,AEL
            WRITE (3) ZVAL,AES
            WRITE (3) ZVAL,AQ
         ELSE
            WRITE (4) ZVAL,AEL
            WRITE (4) ZVAL,AES
            WRITE (4) ZVAL,AQ
         ENDIF
         CALL SHFT(AEL,NY,NT)
         CALL SHFT(AES,NY,NT)
         NWRT=NWRT+3
      ENDIF
C
C --- ENTER THE LOOP OVER STEPS IN Z
C
      DO 500 I0=1,NST
      CALL SECOND(SSTEP1)
C
C - EXIT CONDITION:  STORAGE IS FILLED
      IF (I0.GT.NMAX) THEN
         WRITE(59,50)
 50      FORMAT(' NMAX REACHED')
```

```
          GO TO 510
       ENDIF
       ZVAL=I0*ZSTEP
C
C -- CALCULATE THE FIRST EULER STEP
C
       CALL SECOND(SFE1)
       CALL DERIV(1)
       CALL SECOND(SFE2)
       DO 100 I2=1,NY
       DO 100 I3=1,NT
       C1=AW(I3,I2,1)
       C2=AW(I3,I2,2)
       AW(I3,I2,1)=AEL(I3,I2)*CYVEC(I2,1)
       AW(I3,I2,2)=AES(I3,I2)*CYVEC(I2,2)
       AEL(I3,I2)=AW(I3,I2,1)+ZH*C1
       AES(I3,I2)=AW(I3,I2,2)+ZH*C2
 100   CONTINUE
C
C - SOLVE CONSTRAINT EQUATION FOR THE MATERIAL EXCITATION
       CALL SECOND(SQ1)
       CALL DETQ(TRAT)
       CALL SECOND(SQ2)
C
C -- CALCULATE THE SECOND EULER STEP
C
       CALL SECOND(SFE3)
       CALL DERIV(2)
       CALL SECOND(SFE4)
       DO 110 I2=1,NY
       DO 110 I3=1,NT
       AEL(I3,I2)=AW(I3,I2,1)*CYVEC(I2,1)+ZSTEP*AW(I3,I2,3)
       AES(I3,I2)=AW(I3,I2,2)*CYVEC(I2,2)+ZSTEP*AW(I3,I2,4)
 110   CONTINUE
C
C - SOLVE CONSTRAINT EQUATION FOR THE MATERIAL EXCITATION
       CALL SECOND(SQ3)
       CALL DETQ(TRAT)
       CALL SECOND(SQ4)
C
C -- RECORD DATA
C
       IF (ZVAL.GE.ZKP) THEN
          ZKP=ZKP+ZKEEP
          IF (NT.GT.8.AND.NY.GT.8) THEN
             IZNO=IZNO+1
             INUMRL=AINT(0.01*IZNO)
             ISTP1=NUMRAL (INUMRL+1:INUMRL+1)
             IRST=IZNO-100*INUMRL
             INUMRL=AINT(0.1*IRST)
             ISTP2=NUMRAL (INUMRL+1:INUMRL+1)
             INUMRL=IRST-10*INUMRL
             ISTP3=NUMRAL (INUMRL+1:INUMRL+1)
             DTFL2D=FRM//'2'
             CALL ASSIGN(IRRE,'DN'L,DTFL2D,'A'L,'FT04'L)
             WRITE (59,*) 'ASSIGN FT04= ',DTFL2D
             PDN2D=FRAM//ISTP1//ISTP2//ISTP3
          ENDIF
          IF (NY.GT.8) THEN
             DO 115 I2=1,NY
             DO 115 I3=1,NT
             EL(I3,I2)=YFAC*EL(I3,I2)
             ES(I3,I2)=YFAC*ES(I3,I2)
```

```
            AQ(I3,I2)=Q(I3,I2)
 115        CONTINUE
            CALL SHFT(EL,NY,NT)
            CALL SHFT(ES,NY,NT)
            CALL SHFT(Q,NY,NT)
            WRITE (4) ZVAL,EL
            WRITE (4) ZVAL,ES
            WRITE (4) ZVAL,Q
            CALL SHFT(EL,NY,NT)
            CALL SHFT(ES,NY,NT)
            CALL SHFT(Q,NY,NT)
            CALL CFOUR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
            CALL SHFT(AEL,NY,NT)
            CALL SHFT(AES,NY,NT)
            CALL SHFT(AQ,NY,NT)
            WRITE (4) ZVAL,AEL
            WRITE (4) ZVAL,AES
            WRITE (4) ZVAL,AQ
            NWRT=NWRT+6
            CALL SHFT(AEL,NY,NT)
            CALL SHFT(AES,NY,NT)
            IF (NT.GT.8) THEN
               CLOSE (4)
               CALL SAVE(IRRE,'DN'L,DTFL2D,'PDN'L,PDN2D,
     1                           'RESIDE'L,'OFFLINE'L)
               CALL RELEASE (IRRE,'DN'L,DTFL2D)
            ENDIF
         ELSE
            WRITE (4) ZVAL,EL
            WRITE (4) ZVAL,ES
            WRITE (4) ZVAL,Q
            NWRT=NWRT+3
         ENDIF
      ENDIF
      CALL SECOND(SSTEP2)
C
C -- SET TIMING DATA
C
      SSTEP(I0)=SSTEP2-SSTEP1
      SFE(I0)=SFE4-SFE3+SFE2-SFE1
      SQ(I0)=SQ4-SQ3+SQ2-SQ1
C
C - EXIT CONDITION:  ZFINAL IS REACHED
      IF (ZVAL.GE.ZFINAL) GO TO 510
 500  CONTINUE
C
C -- EXIT ROUTINES
C
 510  CONTINUE
C
C - SET STOT; RECORD CPU TIMING DATA
      CALL SECOND(STOT2)
      STOT=STOT2-STOT1
      NWRT=NWRT+1
      IF (NT.GT.8.AND.NY.GT.8) THEN
         WRITE (3) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
         CLOSE (3)
         CALL SAVE(IRRE,'DN'L,DTFL1,'PDN'L,PDN1,'RESIDE'L,'OFFLINE'L)
      ELSE
         WRITE (4) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
         CLOSE (4)
         CALL SAVE(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D,'RESIDE'L,'OFFLINE'L)
      ENDIF
      CALL EXIT(1)
      END
C
C
C
C
C
      SUBROUTINE DETQ(TRAT)
C
C  THIS SUBROUTINE FIRST CALCULATES THE PUMP AND AND STOKES FIELDS IN
C  THE COORDINATE SPACE FROM THEIR FOURIER SPACE REPRESENTATIONS.
C  IT THEN DETERMINES THE MATERIAL EXCITATION (Q) IN THREE DIFFERENT
C  PARAMETER REGIMES:  1) IF NT IS LESS THAN OR EQUAL TO 8, A SET OF
C  1-D STATIONARY CASES IS RUN.  2) IF MAX(ABS(T/TTWO)) < 10.0 AND
C  NT > 8, A RUNNING SUM IS PERFORMED.  3) IF MAX(T/TTWO) > 10.0
C  AND NT > 8, AN FFT APPROACH IS USED.
C
C                        --VARIABLES--
C
C      TRAT = MAX(T/TTWO)
C
      PARAMETER(NT=256,NY=256,NTHP=1+NT/2,NS=5*NY/2)
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
     1 AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),COMVEC(NT),CWQ(NT),
     2 WQ1(NT),WQ2(NT)
      EQUIVALENCE (CWQ,WQ1),(CWQ(NTHP),WQ2)
      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
      COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT
C
      DO 10 I2=1,NY
      DO 10 I3=1,NT
      EL(I3,I2)=AEL(I3,I2)
      ES(I3,I2)=AES(I3,I2)
   10 CONTINUE
      IF (NY.GT.8) THEN
         CALL CFOUR2(EL,CW,NY,NT,1,1,AW1,AW2)
         CALL CFOUR2(ES,CW,NY,NT,1,1,AW1,AW2)
      ENDIF
      IF (NT.LE.8) THEN
         DO 20 I2=1,NY
         DO 20 I3=1,NT
   20    Q(I3,I2)=-(0.0,1.0)*RKAP1*TTWO*CONJG(ES(I3,I2))*EL(I3,I2)
      ELSEIF (TRAT.LE.10.0) THEN
         DO 30 I2=1,NY
         DO 30 I3=2,NT
   30    Q(I3,I2)=Q(I3-1,I2)-(0.0,1.0)*RKAP1*RDT*CONJG(ES(I3,I2))
     1     *EL(I3,I2)*WQ1(I3)
         DO 35 I2=1,NY
         DO 35 I3=1,NT
   35    Q(I3,I2)=WQ2(I3)*Q(I3,I2)
      ELSE
         DO 40 I2=1,NY
         DO 40 I3=1,NT
   40    Q(I3,I2)=CONJG(ES(I3,I2))*EL(I3,I2)
         CALL INVERT(Q,AQ,NT,NY)
         CALL CFOUR2(AQ,CWQ,NT,NY,1,1,AW1,AW2)
         DO 45 I3=1,NT
         DO 45 I2=1,NY
   45    AQ(I2,I3)=COMVEC(I3)*AQ(I2,I3)
         CALL CFOUR2(AQ,CWQ,NT,NY,-1,1,AW1,AW2)
         CALL INVERT(AQ,Q,NY,NT)
      ENDIF
      IF (NY.GT.8) THEN
         R1=YFAC**2
         DO 50 I2=1,NY
         DO 50 I3=1,NT
 50      Q(I3,I2)=R1*Q(I3,I2)
      ENDIF
      RETURN
      END
C
C
C
C
C
      SUBROUTINE DERIV(IFILL)
C
C  THIS SUBROUTINE CALCULATES THE Z-DERIVATIVES OF THE PUMP AND STOKES
C  FIELDS.  THIS CALCULATION IS DONE IN KY-SPACE.  THE LINEAR PORTION OF
C  THE SECOND-ORDER-DERIVATIVE OPERATOR HAS A FINITE STEP CORRECTION
C  (CONTAINED IN CYVEC) SO THAT THE LINEAR CONTRIBUTION IS EXACT.
C
C                               —VARIABLES—
C
C       IFILL = DERIVATIVE NUMBER
C             = 1:  INITIAL STEP
C             = 2:  MID-POINT STEP
C
      PARAMETER(NT=256,NY=256,NS=5*NY/2)
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),AEL(NT,NY),AES(NT,NY),
     1 AQ(NT,NY),AW(NT,NY,4),CW(NS),AW1(NY),AW2(NY),COMVEC(NT),CWQ(NT)
      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
      COMMON/VWORK/AEL,AES,AQ,AW,CWQ,COMVEC,RKAP1,RKAP2,TTWO,YFAC,RDT
C
      C1=-(0.0,1.0)*(RKP/RKS)*RKAP2
      IF (NY.GT.8) C1=C1/NY
      DO 10 I2=1,NY
      DO 10 I3=1,NT
 10   AQ(I3,I2)=C1*Q(I3,I2)*ES(I3,I2)
      IF (NY.GT.8) CALL CFOUR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
      IV=2*IFILL-1
      DO 20 I2=1,NY
      DO 20 I3=1,NT
 20   AW(I3,I2,IV)=AQ(I3,I2)
      C1=-(0.0,1.0)*RKAP2
      IF (NY.GT.8) C1=C1/NY
      DO 30 I2=1,NY
      DO 30 I3=1,NT
 30   AQ(I3,I2)=C1*CONJG(Q(I3,I2))*EL(I3,I2)
      IF (NY.GT.8) CALL CFOUR2(AQ,CW,NY,NT,-1,1,AW1,AW2)
      IV=2*IFILL
      DO 40 I2=1,NY
      DO 40 I3=1,NT
 40   AW(I3,I2,IV)=AQ(I3,I2)
      RETURN
      END
C
C
C
C
      SUBROUTINE SHFT(FDATA,NF,NV)
C
C  THIS SUBROUTINE SHIFTS THE FOURIER DATA SO THAT ZERO FREQUENCY IS AT
C  THE 1+NF/2 LOCATION (THE CENTER OF THE ARRAY)
```

```
C
      DIMENSION FDATA(2*NV,NF)
      NFH=NF/2
      DO 100 I2=1,NFH
      DO 100 I3=1,2*NV
      TEMP=FDATA(I3,I2)
      FDATA(I3,I2)=FDATA(I3,I2+NFH)
      FDATA(I3,I2+NFH)=TEMP
 100  CONTINUE
      RETURN
      END
C
C
C
C
      SUBROUTINE INVERT(EDATA,EWORK,NF,NV)
C
C  THIS SUBROUTINE INVERTS THE INNER AND OUTER ARRAY VARIABLES
C
      COMPLEX EDATA(NF,NV),EWORK(NV,NF)
      IF (NF.LE.1.OR.NV.LE.1) RETURN
      DO 50 I3=1,NF
      DO 50 I2=1,NV
      EWORK(I2,I3)=EDATA(I3,I2)
 50   CONTINUE
      RETURN
      END
C
C
C
C
       SUBROUTINE INIT(ICOND)
C
C  THIS SUBROUTINE DETERMINES THE INITIAL PROFILES FOR THE STOKES AND
C  PUMP WAVES.  MOST VARIABLES ARE DECLARED IN THE MAIN ROUTINE.
C
C                        --VARIABLES--
C
C      ICOND = 1:  DOUBLE-SECH PROFILE
C              2:  SECH**2-HYPERGAUSSIAN PROFILE WITH STOKES ASYMMETRY
C                  IN TIME AND IMPOSED CHIRP
C              3:  1-D TRANSIENT PROFILES:  TYPE DETERMINED BY ITYPE
C                  ITYPE = 1:  SECH**N PROFILES
C                          2:  RECTANGULAR PROFILES
C                          3:  LORENTZIAN**N PROFILES
C                          4:  EXP(|AT|**N) PROFILES
C              4:  STATIONARY HYPERGAUSSIAN PROFILES WITH PUMP
C                  ABERRATION INCLUDED
C
      PARAMETER(NT=256,NY=256,NP=10,NYH=NY/2,NYHP=NYH+1,NS=5*NY/2)
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION EL(NT,NY),ES(NT,NY),Q(NT,NY),CW(NS),AW1(NY),AW2(NY),
     1 YSTOR1(NY),YSTOR2(NY),YSC(NY),TSTORE(NT),TSC(NT),PHL(NP),
     2 YWIDTH(NP),TWIDTH(NP),YOFF(NP),TOFF(NP),RAMP(NP),ITYPE(8),
     3 RTYPE(8),RABAMP(8),RDSLIM(8),YM(2),TM(2)
      COMMON/VINIT/NPUMP,YM,TM,ZINT,YOFF,TOFF,YWIDTH,TWIDTH,YOST,TOST,
     1 YWST,TWST,RAMP,RAST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,ITYPE,RTYPE,
     2 RABAMP,RDSLIM
      COMMON/VARTWO/EL,ES,Q,AW1,AW2,CW,RKP,RKS
      DATA PI/3.14159265358979/,SQ2/1.41421356237309/,
     2 SQ4/1.18920711500272/,RAL2/0.693147180559945/,
     3 SQ10/3.16227766016838/,SQ12/3.46410161513775/
C
```

```
      IF (ICOND.LT.1.OR.ICOND.GT.4) THEN
         WRITE (59,5)
 5       FORMAT(' ONLY TYPES 1,2,3 AND 4 ARE INITIALIZED')
         CALL EXIT(1)
      ENDIF
C
C - INITIALIZE VARIABLE ARRAYS
      DO 8 I2=1,NY
      DO 8 I3=1,NT
      EL(I3,I2)=(0.0,0.0)
      ES(I3,I2)=(0.0,0.0)
      Q(I3,I2)=(0.0,0.0)
 8    CONTINUE
      IF (ICOND.NE.3.AND.NY.LE.8) THEN
         WRITE (59,10)
 10      FORMAT(' ICOND MUST EQUAL 3 IN 1-D TRANSIENT RUNS (NY = 8 OR',
     1     ' LESS):'/,' ICOND IS RESET TO 3')
         ICOND=3
      ENDIF
      IF (ICOND.NE.4.AND.NT.LE.8) THEN
         WRITE (59,12)
 12      FORMAT(' ICOND MUST EQUAL 4 IN STATIONARY RUNS (NT = 8 OR',
     1     ' LESS):'/,' ICOND IS RESET TO 4')
         ICOND=4
      ENDIF
      IF (ICOND.EQ.3) GO TO 210
C
C - INITIALIZE Y-QUANTITIES
      IF (ABS(YM(2)+YM(1)).GT.1.0E-06) WRITE (59,14)
 14   FORMAT(' YM(2) MUST EQUAL -YM(1)')
      YM(2)=-YM(1)
      RDY=(YM(2)-YM(1))/NY
      DO 16 I2=1,NYH
 16   YSTOR2(I2)=RDY*(I2-1)
      DO 18 I2=NYHP,NY
 18   YSTOR2(I2)=RDY*(I2-1-NY)
      IF (ICOND.EQ.4) GO TO 310
C
      RDT=(TM(2)-TM(1))/NT
      IF (ICOND.EQ.2) GO TO 110
      RFAC=2.0*ALOG(1.0+SQ2)
C
C
C --- DOUBLE-SECH PROFILE
C
C
C - DETERMINE PUMP FACTORS
      DO 50 I1=1,NPUMP
      ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
      YFAC=RFAC/YWIDTH(I1)
      TFAC=RFAC/TWIDTH(I1)
      DO 20 I2=1,NY
      Y1=YSTOR2(I2)
      YV=EXP(YFAC*(Y1-YOFF(I1)))
      YSTOR1(I2)=(1.0/(YV+1.0/YV))
 20   CONTINUE
      T1=TM(1)-TOFF(I1)
      DO 30 I3=1,NT
      TV=EXP(TFAC*(T1+RDT*(I3-1)))
      TSTORE(I3)=4.0*RAMP(I1)/(TV+1.0/TV)
 30   CONTINUE
      DO 50 I2=1,NY
      C1=CEXP(ALPHA*YSTOR2(I2))
      DO 50 I3=1,NT
      EL(I3,I2)=EL(I3,I2)+YSTOR1(I2)*TSTORE(I3)*C1
 50   CONTINUE
C
C - DETERMINE STOKES FACTORS
      ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
      YFAC=RFAC/YWST
      DO 70 I2=1,NY
      Y1=YSTOR2(I2)
      YV=EXP(YFAC*(Y1-YOST))
      YSTOR1(I2)=(1.0/(YV+1.0/YV))
 70   CONTINUE
      T1=TM(1)-TOST
      DO 80 I3=1,NT
      TV=EXP(TFAC*(T1+RDT*(I3-1)))
      TSTORE(I3)=4.0*RAST/(TV+1.0/TV)
 80   CONTINUE
      DO 100 I2=1,NY
      C1=CEXP(ALPHA*YSTOR2(I2))
      DO 100 I3=1,NT
      ES(I3,I2)=YSTOR1(I2)*TSTORE(I3)*C1
 100  CONTINUE
      RETURN
C
C
C ---- SECH**2-HYPERGAUSSIAN PROFILE WITH ASYMMETRIC STOKES WAVE
C
 110  CONTINUE
      RFACY=2.0**(NHYP-1)*RAL2
      RFACT=2.0*ALOG(SQ4+SQRT(SQ2-1.0))
C
C - DETERMINE PUMP FACTORS
      DO 150 I1=1,NPUMP
      ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
      YFAC=RFACY/YWIDTH(I1)**NHYP
      TFAC=RFACT/TWIDTH(I1)
      DO 120 I2=1,NY
      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOFF(I1))**NHYP)
 120  CONTINUE
      T1=TM(1)-TOFF(I1)
      DO 130 I3=1,NT
      TV=EXP(TFAC*(T1+RDT*(I3-1)))
      TSTORE(I3)=4.0/(TV+1.0/TV)**2
 130  CONTINUE
      DO 150 I2=1,NY
      DO 150 I3=1,NT
      R1=YSTOR1(I2)*TSTORE(I3)
      EL(I3,I2)=EL(I3,I2)+RAMP(I1)*R1*CEXP((0.0,1.0)*PHL(I1)*R1**2
     1 + ALPHA*YSTOR2(I2))
 150  CONTINUE
C
C - DETERMINE STOKES CHIRP FACTORS
C   AT PRESENT, TWC=TWIDTH(1), YWC=YWIDTH(1)
      TWC=TWIDTH(1)
      YWC=YWIDTH(1)
      YFAC=RFACY/YWC**NHYP
      TFAC=RFACT/TWC
      DO 160 I2=1,NY
      YSC(I2)=EXP(-YFAC*YSTOR2(I2)**NHYP)
 160  CONTINUE
      T1=TM(1)-TOST-TOC
      DO 165 I3=1,NT
      TV=EXP(TFAC*(T1+RDT*(I3-1)))
      TSC(I3)=4.0/(TV+1.0/TV)**2
  165 CONTINUE
C
C - DETERMINE STOKES FACTORS
      ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
      YFAC=RFACY/YWST**NHYP
      TFAC=RFACT/TWST
      DO 170 I2=1,NY
      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOST)**NHYP)
  170 CONTINUE
      DO 180 I3=1,NT
      T1=TM(1)+RDT*(I3-1)-TOST
C
C - SET STOKES ASYMMETRY
      TV=EXP(-TFAC*RALASM*T1)
      T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
      TV=EXP(TFAC*T1)
      TSTORE(I3)=4.0/(TV+1.0/TV)**2
  180 CONTINUE
      DO 190 I2=1,NY
      DO 190 I3=1,NT
      R1=YSTOR1(I2)*TSTORE(I3)
      R2=YSC(I2)*TSC(I3)
      ES(I3,I2)=RAST*R1*CEXP((0.0,1.0)*PHST*R2**2
     1 +ALPHA*YSTOR2(I2))
  190 CONTINUE
      RETURN
C
C
C ---- ONE-DIMENSIONAL TRANSIENT CASES (NO Y-VARIATION)
C
  210 CONTINUE
      IF (NY.GT.8) THEN
         WRITE (59,212)
  212    FORMAT(' IN TRANSIENT STUDIES, ONLY UP TO 8 CASES CAN BE KEPT')
         CALL EXIT(1)
      ENDIF
      RDT=(TM(2)-TM(1))/NT
C
C --- LOOP OVER CASES
C
      DO 290 I1=1,NY
C
C --- SECH PROFILE
C
      IF (ITYPE(I1).EQ.1) THEN
         R2=1.0/RTYPE(I1)
         R1=0.5*R2
         RFACT=2.0*ALOG(EXP(R1*RAL2)+SQRT(EXP(R2*RAL2)-1.0))
C
C - DETERMINE PUMP PROFILE
         TFAC=RFACT/TWIDTH(I1)
         T1=TM(1)-TOFF(I1)
         DO 215 I3=1,NT
         TV=EXP(TFAC*(T1+RDT*(I3-1)))
         TV=2.0/(TV+1.0/TV)
         TV=EXP(RTYPE(I1)*ALOG(TV))
         EL(I3,I1)=RAMP(I1)*TV*CEXP((0.0,1.0)*PHL(I1)*TV**2)
  215    CONTINUE
C
C - DETERMINE STOKES CHIRP FACTOR
         TWC=TWIDTH(1)
         TFAC=RFACT/TWC
         T1=TM(1)-TOST-TOC
         DO 220 I3=1,NT
         TV=EXP(TFAC*(T1+RDT*(I3-1)))
         TV=2.0/(TV+1.0/TV)
         TSC(I3)=EXP(RTYPE(I1)*ALOG(TV))
 220     CONTINUE
C
C - DETERMINE STOKES PROFILE
         TFAC=RFACT/TWST
         DO 225 I3=1,NT
         T1=TM(1)+RDT*(I3-1)-TOST
         TV=EXP(-TFAC*RALASM*T1)
         T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
         TV=EXP(TFAC*T1)
         TV=2.0/(TV+1.0/TV)
         TV=EXP(RTYPE(I1)*ALOG(TV))
         ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
 225     CONTINUE
C
C -- RECTANGULAR PROFILE (ASSYMETRY AND CHIRP ARE IGNORED)
C
      ELSEIF (ITYPE(I1).EQ.2) THEN
C
C - DETERMINE PUMP PROFILE
         IMIN=NINT((TOFF(I1)-TM(1)-0.5*TWIDTH(I1))/RDT) + 1
         IF (IMIN.LT.1) IMIN=1
         IMAX=NINT((TOFF(I1)-TM(1)+0.5*TWIDTH(I1))/RDT) + 1
         IF (IMAX.GT.NT) IMAX=NT
         DO 230 I3=IMIN,IMAX
 230     EL(I3,I1)=RAMP(I1)
C
C - DETERMINE STOKES PROFILE
         IMIN=NINT((TOST-TM(1)-0.5*TWST)/RDT) + 1
         IF (IMIN.LT.1) IMIN=1
         IMAX=NINT((TOST-TM(1)+0.5*TWST)/RDT) + 1
         IF (IMAX.GT.NT) IMAX=NT
         DO 235 I3=IMIN,IMAX
 235     ES(I3,I1)=RAST
C
C -- LORENTZIAN PROFILE
C
      ELSEIF (ITYPE(I1).EQ.3) THEN
         RFACT=2.0*SQRT(EXP((0.5/RTYPE(I1))*RAL2)-1.0)
C
C - DETERMINE PUMP PROFILE
         TFAC=RFACT/TWIDTH(I1)
         T1=TM(1)-TOFF(I1)
         DO 240 I3=1,NT
         TV=T1+RDT*(I3-1)
         TV=1.0/(1.0+(TFAC*TV)**2)
         TV=EXP(RTYPE(I1)*ALOG(TV))
         EL(I3,I1)=RAMP(I1)*TV*CEXP((0.0,1.0)*PHL(I1)*TV**2)
 240     CONTINUE
C
C - DETERMINE STOKES CHIRP FACTOR
         TWC=TWIDTH(1)
         TFAC=RFACT/TWC
         T1=TM(1)-TOST-TOC
         DO 245 I3=1,NT
         TV=T1+RDT*(I3-1)
         TV=1.0/(1.0+(TFAC*TV)**2)
         TSC(I3)=EXP(RTYPE(I1)*ALOG(TV))
 245     CONTINUE
```

```
C
C - DETERMINE STOKES PROFILE
          TFAC=RFACT/TWST
          DO 250 I3=1,NT
          T1=TM(1)+RDT*(I3-1)-TOST
          TV=EXP(-TFAC*RALASM*T1)
          T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
          TV=1.0/(1.0+(TFAC*T1)**2)
          TV=EXP(RTYPE(I1)*ALOG(TV))
          ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
 250      CONTINUE
C
C -- EXPONENTIAL PROFILE (EXPONENT IS TAKEN TO THE POWER RTYPE(I1))
C
       ELSEIF (ITYPE(I1).EQ.4) THEN
          RFACT=2.0*EXP((1.0/RTYPE(I1))*ALOG(0.5*RAL2))
C
C - DETERMINE PUMP PROFILE
          TFAC=RFACT/TWIDTH(I1)
          T1=TM(1)-TOFF(I1)
          DO 255 I3=1,NT
          TV=ABS(TFAC*(T1+RDT*(I3-1)))+1.0E-10
          TV=EXP(-EXP(RTYPE(I1)*ALOG(TV)))
          EL(I3,I1)=RAMP(I1)*TV*EXP((0.0,1.0)*PHL(I1)*TV**2)
 255      CONTINUE
C
C - DETERMINE STOKES CHIRP FACTOR
          TWC=TWIDTH(1)
          TFAC=RFACT/TWC
          T1=TM(1)-TOST-TOC
          DO 260 I3=1,NT
          TV=ABS(TFAC*(T1+RDT*(I3-1)))+1.0E-10
          TSC(I3)=EXP(-EXP(RTYPE(I1)*ALOG(TV)))
 260      CONTINUE
C
C - DETERMINE STOKES PROFILE
          TFAC=RFACT/TWST
          DO 265 I3=1,NT
          T1=TM(1)+RDT*(I3-1)-TOST+1.0E-10
          TV=EXP(-TFAC*RALASM*T1)
          T1=T1*(1.0+RAMASM*TV/(TV+1.0/TV))
          TV=EXP(-EXP(RTYPE(I1)*ALOG(ABS(TFAC*T1))))
          ES(I3,I1)=RAST*TV*CEXP((0.0,1.0)*PHST*TSC(I3)**2)
 265      CONTINUE
C
C -- ERROR
C
       ELSE
          WRITE (59,270) I1,ITYPE(I1)
 270      FORMAT(' ONLY TRANSIENT TYPES 1-4 ARE INITIALIZED'/,
     1    ' ON PUMP NO.',I4,'   TYPE NO. =',I4)
       ENDIF
 290   CONTINUE
       RETURN
C
C
C ---  STATIONARY CASE (NO T-VARIATION)
C      (AT PRESENT THE DISPERSION LIMIT IS SET WITH RESPECT TO YWIDTH(1))
C
 310   CONTINUE
       RFACY=2.0**(NHYP-1)*RAL2
       RFACK=(0.125*YWIDTH(1)**2/(EXP((2.0/NHYP)*ALOG(RAL2))))
     1 *(2.0*PI/(NY*RDY))**2
```

```
C
C - DETERMINE PUMP FACTORS
      DO 320 I1=1,NPUMP
      ALPHA=-(0.0,1.0)*YOFF(I1)*RKP/ZINT
      YFAC=RFACY/YWIDTH(I1)**NHYP
      DO 315 I2=1,NY
      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOFF(I1))**NHYP)
  315 CONTINUE
      DO 320 I3=1,NT
      DO 320 I2=1,NY
      EL(I3,I2)=EL(I3,I2)+RAMP(I1)*YSTOR1(I2)*CEXP(ALPHA*YSTOR2(I2))
  320 CONTINUE
C
C -- DETERMINE RANDOMIZING FACTORS
C
      DO 350 I3=1,NT
      IF (RDSLIM(I3).LT.1.1) GO TO 350
      RKFAC=RFACK/(2.0*RDSLIM(I3)**2-1.0)
C
C - PHASE FACTORS
      DO 322 I2=1,NY
  322 AW1(I2)=CEXP((0.0,1.0)*2.0*PI*RANF(1))
      CALL CFFT2(0,1,NY,AW1,CW,AW2)
      DO 326 I2=1,NYH
      AW2(I2)=EXP(-RKFAC*(I2-1)**2)*AW2(I2)
      AW2(NYH+I2)=EXP(-RKFAC*(NYH-I2+1)**2)*AW2(NYH+I2)
  326 CONTINUE
      CALL CFFT2(0,-1,NY,AW2,CW,AW1)
      DO 330 I2=1,NY
      R1=CABS(AW1(I2))
      IF (R1.GT.1.0E-10) AW1(I2)=AW1(I2)/R1
  330 EL(I3,I2)=EL(I3,I2)*AW1(I2)
C
C - AMPLITUDE FACTORS
      IF (RABAMP(I3).LT.0.01) GO TO 350
      DO 332 I2=1,NY
  332 AW1(I2)=-5.0
      DO 334 J2=1,10
      DO 334 I2=1,NY
  334 AW1(I2)=AW1(I2)+RANF(1)
      R1=0.0
      DO 335 I2=1,NY
  335 R1=R1+AW1(I2)*AW1(I2)
      CALL CFFT2(0,1,NY,AW1,CW,AW2)
      DO 336 I2=1,NYH
      AW2(I2)=EXP(-RKFAC*(I2-1)**2)*AW2(I2)
      AW2(NYH+I2)=EXP(-RKFAC*(NYH-I2+1)**2)*AW2(NYH+I2)
  336 CONTINUE
      CALL CFFT2(0,-1,NY,AW2,CW,AW1)
      R2=0.0
      DO 337 I2=1,NY
      AW1(I2)=AW1(I2)/NY
  337 R2=R2+AW1(I2)*AW1(I2)
      R1=SQRT(R1/R2)*RABAMP(I3)*SQ12/SQ10
      R2=1.0-RABAMP(I3)
      DO 340 I2=1,NY
  340 EL(I3,I2)=EL(I3,I2)*(R2+R1*AW1(I2))
  350 CONTINUE
C
C - DETERMINE STOKES PROFILE
      ALPHA=-(0.0,1.0)*YOST*RKS/ZINT
      YFAC=RFACY/YWST**NHYP
      DO 370 I2=1,NY
      YSTOR1(I2)=EXP(-YFAC*(YSTOR2(I2)-YOST)**NHYP)
 370  CONTINUE
      DO 390 I3=1,NT
      DO 390 I2=1,NY
      ES(I3,I2)=RAST*YSTOR1(I2)*CEXP(ALPHA*YSTOR2(I2))
 390  CONTINUE
      RETURN
C
      END
C
C
C
C
      SUBROUTINE CFOUR2(FDATA,RWORK,NF,NV,ISIGN,ITYPE,RW1,RW2)
C
C  THIS SUBROUTINE WAS WRITTEN BY CURTIS R. MENYUK 11/86.  IT
C  CALCULATES THE FOURIER TRANSFORM OF A SET OF VECTORS STORED IN A TWO-
C  DIMENSIONAL ARRAY.  THE ROUTINE TRANSFORMS OVER THE OUTER VARIABLE
C  (SLOWLY VARYING) AND VECTORIZES OVER THE INNER VARIABLE (RAPIDLY
C  VARYING).  THE ALGORITHM USED IS DESCRIBED IN NUMERICAL RECIPES
C  BY PRESS, ET AL., CHAP. 12.  THE ROUTINE HERE IS BASED ON FOUR1.
C ---------------------------------------------------------------------
C  MODIFIED 5/87:
C  IF NV=8 OR LESS THIS SUBROUTINE USES THE OMNILIB ROUTINE CFFT2
C  TO CARRY OUT THE FOURIER TRANSFORM SERIALLY.
C
C                                   --VARIABLES--
C
C        FDATA = DATA ARRAY.  IN THIS PROGRAM, IT IS TREATED AS A REAL
C                ARRAY WITH 2*NV X NF ELEMENTS.  THE CORRESPONDING
C                COMPLEX ARRAY HAS NV X NF ELEMENTS.
C        RWORK = WORK ARRAY WHERE THE NEEDED COSINES AND SINES ARE STORED.
C                IT HAS 2*(NF-1) ELEMENTS
C        NF = THE OUTER DIMENSION OVER WHICH THE ROUTINE TRANSFORMS
C        NV = THE INNER DIMENSION OVER WHICH THE PROGRAM VECTORIZES
C        ISIGN = SIGN OF THE FOURIER TRANSFORM
C        ITYPE = 0:  INITIALIZE THE WORK ARRAY (NF IS THE ONLY SIGNIFICANT
C                    PARAMETER; NV AND ISIGN ARE IGNORED)
C                1:  CARRY OUT THE FOURIER TRANSFORM
C        RW1,RW2 = WORK ARRAYS WITH 2*NF ELEMENTS USED BY CFFT2
C                  (INACTIVE WHEN NV > 8)
C        ICR,ICI = REFERENCES TO THE WORK ARRAY
C        MMAX = SUMMATION SEPARATION IN THE DANIELSON-LANCZOS ROUTINE
C
      DATA TWOPI/6.28318530717959/
      DIMENSION FDATA(2*NV,NF),RWORK(5*NF/2),RW1(2*NF),RW2(2*NF)
      IER=-1
      IF (ITYPE.EQ.0) GO TO 1000
      IF (ITYPE.NE.1) THEN
C
C - ERROR CHECK
         IER=-1
         RETURN
      ENDIF
C
C -- IF NV = 8 OR LESS, CALCULATE FOURIER TRANSFORM SERIALLY
C
      IF (NV.LE.8) THEN
         DO 20 I3=1,NV
         DO 10 I2=1,NF
         RW1(2*I2)=FDATA(2*I3,I2)
         RW1(2*I2-1)=FDATA(2*I3-1,I2)
 10      CONTINUE
```

```
         CALL CFFT2(0,ISIGN,NF,RW1,RWORK,RW2)
         DO 20 I2=1,NF
         FDATA(2*I3,I2)=RW2(2*I2)
         FDATA(2*I3-1,I2)=RW2(2*I2-1)
 20      CONTINUE
         RETURN
      ENDIF
C
C -- NV > 8
C
C -- BIT REVERSAL ROUTINE
C
      J=1
      DO 100 I=1,NF
      IF (J.GT.I) THEN
         DO 40 I1=1,2*NV
         TEMP=FDATA(I1,J)
         FDATA(I1,J)=FDATA(I1,I)
         FDATA(I1,I)=TEMP
 40      CONTINUE
      ENDIF
      M=NF/2
 50   CONTINUE
      IF ((M.GE.1).AND.(J.GT.M)) THEN
         J=J-M
         M=M/2
         GO TO 50
      ENDIF
      J=J+M
 100  CONTINUE
C
C -- DANIELSON-LANCZOS ROUTINE.  THE FOURIER DATA IS RECOMBINED.
C
      MMAX=1
      ICR=1
      ICI=2
      FSIGN=FLOAT(ISIGN)
 120  CONTINUE
      IF (NF.GT.MMAX) THEN
         ISTEP=2*MMAX
         DO 200 M=1,MMAX
         DO 180 I=M,NF,ISTEP
         J=I+MMAX
         DO 180 I1=1,2*NV,2
         TEMPR=RWORK(ICR)*FDATA(I1,J)-FSIGN*RWORK(ICI)*FDATA(I1+1,J)
         TEMPI=RWORK(ICR)*FDATA(I1+1,J)+FSIGN*RWORK(ICI)*FDATA(I1,J)
         FDATA(I1,J)=FDATA(I1,I)-TEMPR
         FDATA(I1+1,J)=FDATA(I1+1,I)-TEMPI
         FDATA(I1,I)=FDATA(I1,I)+TEMPR
         FDATA(I1+1,I)=FDATA(I1+1,I)+TEMPI
 180     CONTINUE
         ICR=ICR+2
         ICI=ICI+2
 200     CONTINUE
         MMAX=ISTEP
         GO TO 120
      ENDIF
      RETURN
C
C
C --- ENTER THE INITIALIZATION ROUTINE
C
 1000 CONTINUE
```

```
C
C — IF NV = 8 OR LESS
C
      IF (NV.LE.8) THEN
         CALL CFFT2(1,1,NF,RW1,RWORK,RW2)
         RETURN
      ENDIF
C
C — IF NV > 8
C
      MMAX=1
      ICR=1
      ICI=2
 1120 CONTINUE
      IF (NF.GT.MMAX) THEN
         ISTEP=2*MMAX
         THETA=TWOPI/ISTEP
         WPR=-2.0*SIN(0.5*THETA)**2
         WPI=SIN(THETA)
         WR=1.0
         WI=0.0
         DO 1200 M=1,MMAX
         RWORK(ICR)=WR
         RWORK(ICI)=WI
         ICR=ICR+2
         ICI=ICI+2
         TEMP=WR
         WR=WR*WPR-WI*WPI+WR
         WI=WI*WPR+TEMP*WPI+WI
 1200    CONTINUE
         MMAX=ISTEP
         GO TO 1120
      ENDIF
      RETURN
      END
```

PRAM1 (version CD)

```
      PROGRAM PRAM1CD
c
c
C  This program was written by Godehard Hilfer (3/87). It generates
C  contour and cross sectional plots from the data generated by the
C  transient RAMAN amplifier code RAM2D1 written by Prof. Curtis R.
C  Menyuk. To execute this program it has to be linked to the
C  DISSPLA graphics package.
c
C  This is version CD which is adapted for the Central Computing
C  Facility Cray computer at the Naval Research Laboratory (Fall
C  1987). This version reads a record at a time and processes the
C  field data contained in it. The field data of the next record
C  that is being read over-writes the previous data in memory such
C  as to minimize the memory requirements and to accommodate large
C  dimensional field data arrays. This process entails large Input/
C  Output transfer costs. Whenever possible, hence, version C should
C  be used which stores all field data of one z-location. This is
C  recommended particularly for one-dimensional transient (ny < 9)
C  operation of the code.
c
C  The program has the following structure:
c
C           |--------------------|      |-------------------------|
C           |        F...        |      |         NPRAM1          |
C           |  input data file   |      |  input parameter file   |
C           |--------------------|      |-------------------------|
C                     .                              .
C                     ................................
C                     .
C                     .
C  |--------------------------------------------------------------------|
C  |                                       PRAM1                        |
C  |                  .              (this program)                     |
C  |     --------------------                                           |
C  |    ||  --------------  ||                                          |
C  |  ..||  |  Main part  | ||                                          |
C  |  . ||  |             | ||- - - - - - - - - - - - -                 |
C  |  . ||  --------------  ||                         |                |
C  |  .  --------------------                          |                |
C  |  .        |        |                              |                |
C  |  .        |        |                              |                |
C  |  .        |        - - - - -                      |                |
C  |  .        |                |                      |                |
C  |  .     ---------           |                  ---------            |
C  |  ....| cntr |              |                  | crssct |.....       |
C  |  .     ---------           |                  ---------     .      |
C  |  .     |    |    |         |                    |      |    .      |
C  |  .     |    |    |         |                    |      |    .      |
C  |  .     |    |    - - - - - nysxis - - - - - - - -      |    .      |
C  |  .     |    |                |                         |    .      |
C  |  .     |    |              powbas                      |    .      |
C  |  .     |    |                                          |    .      |
C  |  .     |    |                                          |    .      |
C  |  .     |    - - - - - - - -xisFFT- - - - - - - - - - - -    .      |
C  |  .     |                                                    .      |
C  |  .     |                                                    .      |
C  |  .   mycon                                                  .      |
C  |  .                                                          .      |
C  |--------------------------------------------------------------------|
C     .                                                          .
C     .                                                          .
```

```
C       .                                                        .
C       :          |-----------------------|                     :
C       ...........|       PLT2.PLT        |.....................
C                  |  graphics output file |
C                  |-----------------------|
c
C     The program starts by setting default values for the graphics output
C     parameters as specified in the data statements. These default values
C     are updated by the values in the input file NPRAM1 which allows
C     format-free input through the two namelists condat and zplot. The
C     updated set is then written, depending on the value of the flag
C     parameters lprmt, onto the first 4 graphics frames in the output file
C     F3RAM00X. The value of lprmt(n) should be equal to 1 if the nth
C     page of parameter output is desired, and equal to 0 if not
c
C     Several constants are precalculated before the large DO-loop 500
C     reads through and plots the data in file F... . Among these constants
C     are the end values and interval sizes for the frequently ploted y
C     and t coordinate axes.
c
C     The following main part of this program acquires the electriC field
C     data from the input data file F... by reading sequentially the i-th
C     record specified by the value i of the consecutive elements of the
C     vector kz. These amplitude data are converted into intensity data
C     if necessary and then handed through the arrays srf and srfi to the
C     subroutine cntr (for contour plotting) and to the subroutine crsect
C     (for cross sectional plots). The sequence of the resulting plots is
C     as follows:
C                  I contours pump intensity
C                      1 sections pump intensity
C                      2 sections pump phase
C                      3 sections pump amplitude (real/imag)
C                 II contours pump FFT intensity
C                      4 sections pump FFT intensity
C                      5 sections pump FFT phase
C                      6 sections pump FFT amplitude (real/imag)
C                III contours Stokes intensity
C                      7 sections Stokes intensity
C                      8 sections Stokes phase
C                      9 sections Stokes amplitude (real/imag)
C                 IV contours Stokes FFT intensity
C                     10 sections Stokes FFT intensity
C                     11 sections Stokes FFT phase
C                     12 sections Stokes FFT amplitude (real/imag)
C                  V contours mat. exct. intensity
C                     13 sections mat. exct. intensity
C                     14 sections mat. exct. phase
C                     15 sections mat. exct. amplitude (real/imag)
C                 VI contours mat. exct. FFT intensity
C                     16 sections mat. exct. FFT intensity
C                     17 sections mat. exct. FFT phase
C                     18 sections mat. exct. FFT amplitude (real/imag)
C                VII contours pump and Stokes intensity
C                     19 sections sum of pump and Stokes intensity
C               VIII contours pump and Stokes FFT intensity
c
C     The roman numerals tell which element of the vector isrf is the
C     flag that determines if that particular contour plot will be done
C     or skipped:
C                    isrf(n) = 0 plot skipped
C                    isrf(n) = 1 plot drawn with labeled contours
C                    isrf(n) =-1 plot drawn; no labels on contours
C     The arabiC numerals of the sections indicate the row of the complex
```

```
C     array cseC  whith which this section is associated. The column number
C     (second index) of the elements of cseC numbers the cross sectional
C     plots of that particular surface. A maximum of nseC (< 9) cross
C     sections of each surface can be drawn. The imaginary part of the
C     elements of cseC determines:  if =0.0 that this sectional plot is
C     not requested
C           if =1.0 that this is a cross section parallel to the y-axis
C                   of the surface under question at a fixed x-value as
C                   given in real units by the real part of the
C                   element of csec; i.e. the first index of the
C                   data array(s) srf(i) is being held constant for this
C                   plot at the value iseC which is the grid point that
C                   corresponds best to the fixed x-value;
C           if =2.0 that this is a cross section parallel to the x-axis
C                   of the surface under question at a fixed y-value as
C                   given in real units by the real part of the element
C                   of cseC in question; i.e. the second index of the
C                   data array(s) srf(i) is being held constant for this
C                   plot at the value iseC which is the grid point that
C                   corresponds best to the fixed y-value.
C     In short: the imaginary part tells which variable to hold constant,
C               and the real part tells at what value (in physical units).
c
C     When one dimensional transient cases (ny.le.8) are being investigated
C     the real part of the element of cseC under question has to be set
C     equal to the number (1.0,through 8.0) of the element of the vector
C     itype in subroutine INIT in the code RAM2D1 in order for the
C     sectional plot to contain the correct data and label of that
C     particular case. Recall that the imaginary part has to be nonzero
C     for the section to be drawn. Note that the sections #19 are sofar
C     intended only for the check of the total electromagnetiC intensity
C     in the one-dimensional transient cases (ny.le.8). Set the real and
C     imaginary part of the elements in row 19 of the array cseC as
C     described above in this paragraph to obtain these total
C     electromagnetiC intensity sectional plots.
c
C     More details on how the individual subroutines work precedes their
C     listings. The contouring subroutine cntr makes use of the
C     subroutine mycon which generates a customized dotted line for the
C     half-height contour. The cross section subroutine crssct calls
C     frequently on the subroutine nysxis which finds 'nice' values for
C     coordinate axis limits and intervals. nysxis in turn uses
C     subroutine powbas to find the next lower integral power of 10 for
C     maximas and minimas. Both subroutines cntr and crssct share
C     subroutine xisFFT when making secondary axes for FFT-plots.
c
C                                 --variables--
c
C         gain = see RAM2D1
C         grfsz = physical size of graphics plots
C         i2 = y-coordinate index in do-loops 125,128,133,136,145,148,153,
C              156,165,168,173,176,185,188,193,196,205,208,213,225,228,
C              233,236,250,260
C         i3 = t-coordinate index in same do-loops as i2
C         icond = see RAM2D1
C         iflip = 0/1 summand checks next row of cseC in do-loops 130,150,
C              170,190,210,230
C         iln = number of dashed contours between solid contours in
C              sub=cntr
C         is = cseC column index in do-loops 120,140,160,180,200,220;
C              dummy index in do-loops 130,150,170,190,210,230
C         ishm = flag for half-height contour option in sub=cntr
C         isrf = flag vector that indicates which contour plots are desired
```

```
C       iss = cseC column index in do-loops 130,150,170,190,210,230
C       jsrf = signed contour plot index; >0 for contour labels, <0 no
C            labels
C       kz = vector contains the iteration numbers at which graphics
C            plots are desired
C       level = vector containing desired level heights for dashed
C            contours
C       lprmt = flag; if nonzero indicates list of parameters is desired
C       necveC = data switch for subroutine nyaxis
C       ndeC = desired number of solid contours representing powers of 10
C       nhyp = see RAM2D1
C       nmax = see RAM2D1; index limit in do-loop 500
C       np = see RAM2D1
C       npump = see RAM2D1
C       nsC = cseC row index in do-loops 130,150,170,190,210,230
C       nseC = number of elements tested in rows of csec
C       nt = see RAM2D1
C       ntp = nt+1
C       nwrt = number of records in unit 4
C       ny = see RAM2D1
C       nyp = ny +1
C       nyh = ny/2
C       nyhp = nyh+1
C       phi = see RAM2D1
C       phst = see RAM2D1
C       pi = 3.14159265358979
C       r1 = intensity normalization factor 8*pi/speed
C       rabamp = see RAM2D1
C       ralasm = see RAM2D1
C       ramasm = see RAM2D1
C       ramp = see RAM2D1
C       rdslim = see RAM2D1
C       rdt = step size in time
C       rdy = step size in transverse spatial variable y
C       rint = see RAM2D1
C       rist = see RAM2D1
C       rkp = see RAM2D1
C       rks = see RAM2D1
C       sC = sum of imaginary parts of a row of csec; test variable
C       sfe = see RAM2D1
C       sinit = see RAM2D1
C       speed = see RAM2D1
C       sq = see RAM2D1
C       srf = array of data from which contours and sections are plotted
C       srfi = imaginary part of amplitude data for cross sectional plots
C       sstep = see RAM2D1
C       stot = see RAM2D1
C       tm = see RAM2D1
C       tm1 = time coordinate lower limit
C       tm2 = time coordinate upper limit
C       tmax = value at end of time axis
C       toC = see RAM2D1
C       toff = see RAM2D1
C       torig = value at beginning of time axis
C       tost = see RAM2D1
C       tstp = time axis interval
C       ttwo = see RAM2D1
C       twidth = see RAM2D1
C       twst = see RAM2D1
C       wfmax = nice spatial FFT axis end value
C       wforig = nice spatial FFT axis beginning value
C       wfstp = nice spatial FFT axis interval
C       yfmax = value at end of spatial FFT axis
```

```
C      yforig = value at beginning of spatial FFT axis
C      yfstp = spatial FFT axis interval
C      ym = see RAM2D1
C      ym1 = y-coordinate lower limit
C      ym2 = y-coordinate upper limit
C      ym2m1 = ym2-ym1
C      ymax = value at end of transverse spatial axis
C      yoff = see RAM2D1
C      yorig = value at beginning of transverse spatial axis
C      yost = see RAM2D1
C      ystp = transverse spatial axis interval
C      ywidth = see RAM2D1
C      ywst = see RAM2D1
C      zfinal = see RAM2D1
C      zint = see RAM2D1
C      zkeep = see RAM2D1
C      zstep = see RAM2D1
C      zval = value of z-coordinate of current data/plot
C *********************************************************************
C MODIFICATION 9/87:
C THE DATA OUTPUT FILE NAME WAS CHANGED FROM 'FRAM' TO THE FOLLOWING:
C THE DATA FILE NAME'S FIRST CHARACTER (F) STANDS FOR THE OLD DATA
C FILE NAME 'FRAM'. THE SECOND CHARACTER INDICATES THE T-DIMENSION,
C THE THIRD THE Y-DIMENSION. THE DIMENSIONS ARE REPRESENTED BY THEIR
C NUMBER (1-8) IF LESS THAN 9. IF GREATER THAN 8 THE DIMENSIONS ARE
C ASSUMED TO BE INTEGRAL POWERS OF 2. THE N-TH POWER OF 2 IS
C REPRESENTED BY THE N-TH CHARACTER OF RLFBET. THE FOURTH THROUGH
C NINETH CHARACTER IN THE FILE NAME ENCODES THE MONTH, DAY, AND YEAR
C THE PROGRAM WAS STARTED. A THENTH THROUGH TWELFTH CHARACTER IS
C APPENDED, NUMBERING THE PARTIAL DATA FILES THAT ARE GENERATED
C WHEN THE PROGRAM RUNS TWO-DIMENSIONALLY.
C *********************************************************************
C
      PARAMETER (NP=10,NST=4000,NT=256,NTP=NT+1,NX=8,NXI=19*NX,NY=128,
     1           NYH=NY/2,NYHP=NYH+1,NYP=NY+1,NZ=20)
C
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION INDEX(NP),ISRF(3),ISTAT(2),ITYPE(8),IWHEN(NYP),
     1          IWORK(257),KZ(NZ),LEVEL(8),LPRMT(4),AEQ(NT,NY),
     2          AER(NT,NY),CSEC(19,NX),PHL(NP),RABAMP(8),RAMP(NP),
     3          RDSLIM(8),RINT(NP),RTYPE(8),SFE(NST),SRF(NTP,NYP),
     4          SRFI(NTP,NYP),SRTYOF(1,NP),SQ(NST),SSTEP(NST),TIK(NY),
     5          TM(2),TOFF(NP),TWIDTH(NP),YM(2),YOFF(NP),YWIDTH(NP)
      CHARACTER*1 D1
      CHARACTER*2 D1A
      CHARACTER*2 D2
      CHARACTER*2 D3
      CHARACTER*7 DTFL1
      CHARACTER*6 DTFL1D
      CHARACTER*7 DTFL2D
      CHARACTER*1 DUM1
      CHARACTER*1 DUM2
      CHARACTER*2 EDN
      CHARACTER*9 FRAM
      CHARACTER*6 FRM
      CHARACTER*1 ISTP1
      CHARACTER*1 ISTP2
      CHARACTER*1 ISTP3
      CHARACTER*10 NUMRAL
      CHARACTER*9 PDN1D
      CHARACTER*12 PDN2D
      CHARACTER*12 PDN0
      CHARACTER*12 PDN1
      CHARACTER*26 RLFBET
      CHARACTER*1 TDIM
      CHARACTER*1 YDIM
      INTEGER DONYET,DAY,YEAR
      NAMELIST /FLDATE/ DONYET,MONTH,DAY,YEAR,IPART,NEDN
      NAMELIST /CONDAT/ ILN,ISHM,LEVEL,LPRMT,NDEC,NSEC,ISRF,CSEC
      NAMELIST /ZPLOT/ KZ
      COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
     1        GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
     2        YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
     3        ZVAL
      COMMON /NUM/ RDT,RDY,RDYF,TM1,TM2,YM1,YM2,YM2M1
      EQUIVALENCE (YOFF,SRTYOF)
C
      DATA PI/3.14159265358979/,SPEED/0.0299779/,
     1     WDLIM/0.632120558828558/
      DATA DONYET/1/,MONTH/09/,DAY/28/,YEAR/87/,IPART/002/,NEDN/01/
      DATA ILN/8/,ISHM/0/,LEVEL/2,3,4,5,6,7,8,9/,LPRMT/4*1/,NDEC/2/,
     1     NSEC/5/,ISRF/8*0/,CSEC/NXI*(0.0,0.0)/,GRFSZ/7.0/
      DATA KZ/NZ*0/
C
      CALL ASSIGN (IRRE,'DN'L,'NPRAM1'L,'A'L,'FT01'L)
      CALL ASSIGN (IRRE,'DN'L,'EPRM'L,'A'L,'FT59'L)
      READ (1,FLDATE)
      WRITE (59,*) 'READ (1,FLDATE)'
      WRITE (59,FLDATE)
      READ (1,CONDAT)
      WRITE (59,*) 'READ (1,CONDAT)'
      WRITE (59,CONDAT)
      READ (1,ZPLOT)
      WRITE (59,*) 'READ (1,ZPLOT)'
      WRITE (59,ZPLOT)
C
C - DETERMINE DATA FILE NAME
      RLFBET='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
C               12345678901234567890123456
      NUMRAL='0123456789'
      IF (NT.GT.8) THEN
         ITDIM=NINT(ALOG(FLOAT(NT))/ALOG(2.0))
         TDIM=RLFBET (ITDIM:ITDIM)
      ELSE
         TDIM=NUMRAL (NT+1:NT+1)
      ENDIF
      IF (NY.GT.8) THEN
         IYDIM=NINT(ALOG(FLOAT(NY))/ALOG(2.0))
         YDIM=RLFBET (IYDIM:IYDIM)
      ELSE
         YDIM=NUMRAL (NY+1:NY+1)
      ENDIF
      WRITE (59,*)'ITDIM= ',ITDIM,' TDIM= ',TDIM
      WRITE (59,*)'IYDIM= ',IYDIM,' YDIM= ',YDIM
      IF (MONTH.GT.0.AND.MONTH.LT.10) THEN
         IDUM1=1
         IDUM2=MONTH+1
         D1=NUMRAL (IDUM2:IDUM2)
      ELSEIF (MONTH.EQ.10) THEN
         IDUM1=2
         IDUM2=1
         D1='A'
      ELSEIF (MONTH.EQ.11) THEN
         IDUM1=2
         IDUM2=2
         D1='B'
      ELSEIF (MONTH.EQ.12) THEN
         IDUM1=2
         IDUM2=3
         D1='C'
      ELSE
         WRITE (59,*) 'MONTH INPUT = ',MONTH,' IS OUT OF RANGE'
         CALL EXIT(1)
      ENDIF
      DUM1=NUMRAL (IDUM1:IDUM1)
      DUM2=NUMRAL (IDUM2:IDUM2)
      D1A=DUM1//DUM2
      IF (DAY.LT.1.OR.DAY.GT.31) THEN
         WRITE (59,*) 'DAY INPUT = ',DAY,' IS OUT OF RANGE'
         CALL EXIT(1)
      ENDIF
      IDUM1=INT(DAY/10)
      IDUM2=DAY-10*IDUM1
      DUM1=NUMRAL (IDUM1+1:IDUM1+1)
      DUM2=NUMRAL (IDUM2+1:IDUM2+1)
      D2=DUM1//DUM2
      IF (YEAR.LT.0.OR.YEAR.GT.99) THEN
         WRITE (59,*) 'YEAR INPUT = ',YEAR,' IS OUT OF RANGE'
         CALL EXIT(1)
      ENDIF
      IDUM1=INT(YEAR/10)
      IDUM2=YEAR-10*IDUM1
      DUM1=NUMRAL (IDUM1+1:IDUM1+1)
      DUM2=NUMRAL (IDUM2+1:IDUM2+1)
      D3=DUM1//DUM2
      FRM='F'//TDIM//YDIM//D1//D2
      FRAM='F'//TDIM//YDIM//D1A//D2//D3
      IDUM1=INT(NEDN/10)
      IDUM2=NEDN-IDUM1*10
      DUM1=NUMRAL (IDUM1+1:IDUM1+1)
      DUM2=NUMRAL (IDUM2+1:IDUM2+1)
      EDN=DUM1//DUM2
      IUNIT=4
      IF (NT.GT.8.AND.NY.GT.8) THEN
         ISTP1=NUMRAL(1:1)
         IF (DONYET.EQ.0) THEN
            ISTP2=ISTP1
         ELSE
            ISTP2=NUMRAL(2:2)
         ENDIF
         DTFL1=FRM//ISTP2
         PDN1=FRAM//ISTP1//ISTP1//ISTP2
         WRITE (59,*)'DTFL1= ',DTFL1
         WRITE (59,*)'PDN1= ',PDN1
         WRITE (59,*)'NEDN,EDN= ',NEDN,EDN
         CALL ACCESS(IRRE,'DN'L,DTFL1,'PDN'L,PDN1,'ED'L,EDN)
         CALL ASSIGN(IRRE,'DN'L,DTFL1,'A'L,'FT03'L)
         ISPCT=1
         IUNIT=3
         IF (DONYET.EQ.0) GO TO 40
      ELSE
         DTFL1D=FRM
         PDN1D=FRAM
         WRITE (59,*)'DTFL1D= ',DTFL1D
         WRITE (59,*)'PDN1D= ',PDN1D
         CALL ACCESS(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D,'ED'L,EDN)
         CALL ASSIGN(IRRE,'DN'L,DTFL1D,'A'L,'FT04'L)
      ENDIF
C
```

```
C -- READ TIMING INFORMATION AND SET OF INPUT PARAMETERS
C
C - SKIP TO EOF IN UNIT IUNIT
      CALL SKIPR(IUNIT,6*NST+3,ISTAT)
      WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),' FILES
     1 IN UNIT ',IUNIT,' .'
C
C - BACKUP ONE RECORD IN UNIT IUNIT
      CALL SKIPR(IUNIT,-1,ISTAT)
      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),'
     1  FILES IN UNIT ',IUNIT,' .'
C
C - READ NUMBER OF RECORDS AND TIMING INFORMATION FROM FILE RAM2D1.FOR
C   AND REWIND DATA FILE
      READ (IUNIT) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ
      WRITE (59,*) 'READ (IUNIT) NWRT,ZVAL,STOT,SINIT,SSTEP,SFE,SQ'
      WRITE (59,*) 'NWRT,ZVAL,STOT,SINIT',NWRT,ZVAL,STOT,SINIT
      WRITE (59,*) 'RAM2D1 RAN ',STOT,' SECONDS.'
      REWIND IUNIT
 40   CONTINUE
C
C - READ CODE INPUT PARAMETER
      READ (IUNIT) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1     YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,
     2     TOC,ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,
     3     NMAX,TTWO,GAIN
      WRITE (59,*) 'READ (IUNIT) NPUMP,YM,TM,ZINT...'
      WRITE(59,*)'NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,YOST,
     1     TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
     2     ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,
     3     GAIN'
      WRITE(59,*) NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,YOST,
     1     TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
     2     ICOND,ITYPE,RTYPE,RABAMP,RDSLIM,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,
     3     GAIN
C
C - ERROR CONDITIONS
      IF (NSEC.LT.1.OR.NSEC.GT.NX) THEN
         WRITE (59,*) 'NSEC = ',NSEC,' IS OUT OF RANGE'
         NSEC=NX
      ENDIF
      IF (NT.LE.8.AND.NY.LE.8) THEN
         WRITE (59,*) 'NT AND NY BOTH LESS THAN 9; STOP'
      ENDIF
      IF (ICOND.NE.3.AND.NY.LE.8) THEN
         WRITE (59,*) 'WHEN ICOND=3 NY MUST BE LESS THAN 9; STOP'
          CALL EXIT(1)
      ENDIF
      IF (ICOND.NE.4.AND.NT.LE.8) THEN
         WRITE (59,*) 'WHEN ICOND=4 NY MUST BE LESS THAN 8; STOP'
          CALL EXIT(1)
      ENDIF
C
C - RENAME CONSTANTS
      TM1=TM(1)
      TM2=TM(2)
      YM1=YM(1)
      YM2=YM(2)
      YM2M1=YM2-YM1
C
C - INITIALIZE DISSPLA GRAPHICS
      CALL COMPRS
C
```

```
C - LIST INPUT PARAMETERS ON 3 GRAPHICS FRAMES UPON REQUEST
C
       IF (LPRMT(1).EQ.0) GO TO 80
C
C - FIRST FRAME OF PARAMETERS
      CALL RESET('ALL')
      CALL AREA2D(8.0,10.5)
      CALL HEIGHT(0.17)
      SLT=2.0
      ZL=10.2
      CALL MESSAG('LIST OF INPUT PARAMETERS$',100,SLT,ZL)
      CALL RESET('HEIGHT')
      SL1=1.8
      SLT=0.0
      ZL=9.5
      CALL MESSAG('ICOND    = $',100,SLT,ZL)
      CALL INTNO(ICOND,SL1,ZL)
      IF (NT.GT.8.AND.NY.GT.8) THEN
         ZL=ZL-0.3
         CALL MESSAG('ILN      = $',100,SLT,ZL)
         CALL INTNO(ILN,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('ISHM     = $',100,SLT,ZL)
         CALL INTNO(ISHM,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('NDEC     = $',100,SLT,ZL)
         CALL INTNO(NDEC,SL1,ZL)
      ENDIF
      IF (ICOND.EQ.2.OR.ICOND.EQ.4) THEN
         ZL=ZL-0.3
         CALL MESSAG('NHYP     = $',100,SLT,ZL)
         CALL INTNO(NHYP,SL1,ZL)
      ENDIF
      ZL=ZL-0.3
      CALL MESSAG('NMAX     = $',100,SLT,ZL)
      CALL INTNO(NMAX,SL1,ZL)
      ZL=ZL-0.3
      CALL MESSAG('NPUMP    = $',100,SLT,ZL)
      CALL INTNO(NPUMP,SL1,ZL)
      NZLT=NZL+1
      ZL=ZL-0.3
      CALL MESSAG('NT       = $',100,SLT,ZL)
      CALL INTNO(NT,SL1,ZL)
      NZLT=NZL+1
      ZL=ZL-0.3
      CALL MESSAG('NY       = $',100,SLT,ZL)
      CALL INTNO(NY,SL1,ZL)
      ZL=ZL-0.3
      CALL MESSAG('GAIN     = $',100,SLT,ZL)
      CALL REALNO(GAIN,105,SL1,ZL)
      IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
         ZL=ZL-0.3
         CALL MESSAG('PHST     = $',100,SLT,ZL)
         CALL REALNO(PHST,105,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('RALASM   = $',100,SLT,ZL)
         CALL REALNO(RALASM,105,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('RAMASM   = $',100,SLT,ZL)
         CALL REALNO(RAMASM,105,SL1,ZL)
      ENDIF
      ZL=ZL-0.3
      CALL MESSAG('RIST     = $',100,SLT,ZL)
      IPLACE=2
      IF (ABS(RIST).GT.9999.0.OR.ABS(RIST).LT.0.01) IPLACE=-2
      CALL REALNO(RIST,IPLACE,SL1,ZL)
      ZL=ZL-0.3
      CALL MESSAG('RKP      = $',100,SLT,ZL)
      IPLACE=2
      IF (ABS(RKP).GT.9999.0.OR.ABS(RKP).LT.0.01) IPLACE=-2
      CALL REALNO(RKP,IPLACE,SL1,ZL)
      ZL=ZL-0.3
      CALL MESSAG('RKS      = $',100,SLT,ZL)
      IPLACE=2
      IF (ABS(RKS).GT.9999.0.OR.ABS(RKS).LT.0.01) IPLACE=-2
      CALL REALNO(RKS,IPLACE,SL1,ZL)
      IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
         ZL=ZL-0.3
         CALL MESSAG('TOC      = $',100,SLT,ZL)
         CALL REALNO(TOC,105,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('TOST     = $',100,SLT,ZL)
         CALL REALNO(TOST,105,SL1,ZL)
      ENDIF
      ZL=ZL-0.3
      CALL MESSAG('TTWO     = $',100,SLT,ZL)
      CALL REALNO(TTWO,105,SL1,ZL)
      IF (NT.GT.8) THEN
         ZL=ZL-0.3
         CALL MESSAG('TWST     = $',100,SLT,ZL)
         CALL REALNO(TWST,105,SL1,ZL)
      ENDIF
      IF (NY.GT.8) THEN
         ZL=ZL-0.3
         CALL MESSAG('YOST     = $',100,SLT,ZL)
         CALL REALNO(YOST,105,SL1,ZL)
         ZL=ZL-0.3
         CALL MESSAG('YWST     = $',100,SLT,ZL)
         CALL REALNO(YWST,105,SL1,ZL)
      ENDIF
      ZL=ZL-0.3
      CALL MESSAG('ZFINAL   = $',100,SLT,ZL)
      CALL REALNO(ZFINAL,105,SL1,ZL)
      IF (NY.GT.8) THEN
         ZL=ZL-0.3
         CALL MESSAG('ZINT     = $',100,SLT,ZL)
         CALL REALNO(ZINT,105,SL1,ZL)
      ENDIF
      ZL=ZL-0.3
      CALL MESSAG('ZKEEP    = $',100,SLT,ZL)
      CALL REALNO(ZKEEP,105,SL1,ZL)
      ZL=ZL-0.3
      CALL MESSAG('ZSTEP    = $',100,SLT,ZL)
      CALL REALNO(ZSTEP,105,SL1,ZL)
      CALL ENDPL(0)
 80   CONTINUE
      IF (LPRMT(2).EQ.0) GO TO 85
C
C - SECOND FRAME OF PARAMETERS
      CALL AREA2D(8.0,10.5)
      ZL=10.2
      CALL MESSAG('LIST OF INPUT PARAMETERS (CONTD)$',100,SLT,ZL)
      SL1=2.2
      SL2=2.9
      SL3=3.6
      SL4=4.3
      SL5=5.0
      SL6=5.7
      SL7=6.4
      SL8=7.1
      ZL=9.5
      IF (NT.GT.8.AND.NY.GT.8) THEN
         CALL MESSAG('ISRF(1-8)  =  $',100,SLT,ZL)
         CALL INTNO(ISRF(1),SL1,ZL)
         CALL INTNO(ISRF(2),SL2,ZL)
         CALL INTNO(ISRF(3),SL3,ZL)
         CALL INTNO(ISRF(4),SL4,ZL)
         CALL INTNO(ISRF(5),SL5,ZL)
         CALL INTNO(ISRF(6),SL6,ZL)
         CALL INTNO(ISRF(7),SL7,ZL)
         CALL INTNO(ISRF(8),SL8,ZL)
         ZL=ZL-0.3
         CALL MESSAG('LEVEL      = $',100,SLT,ZL)
         CALL INTNO(LEVEL(1),SL1,ZL)
         CALL INTNO(LEVEL(2),SL2,ZL)
         CALL INTNO(LEVEL(3),SL3,ZL)
         CALL INTNO(LEVEL(4),SL4,ZL)
         CALL INTNO(LEVEL(5),SL5,ZL)
         CALL INTNO(LEVEL(6),SL6,ZL)
         CALL INTNO(LEVEL(7),SL7,ZL)
         CALL INTNO(LEVEL(8),SL8,ZL)
      ENDIF
      IF (ICOND.EQ.3) THEN
      ZL=ZL-0.3
         CALL MESSAG('ITYPE      = $',100,SLT,ZL)
         CALL INTNO(ITYPE(1),SL1,ZL)
         CALL INTNO(ITYPE(2),SL2,ZL)
         CALL INTNO(ITYPE(3),SL3,ZL)
         CALL INTNO(ITYPE(4),SL4,ZL)
         CALL INTNO(ITYPE(5),SL5,ZL)
         CALL INTNO(ITYPE(6),SL6,ZL)
         CALL INTNO(ITYPE(7),SL7,ZL)
         CALL INTNO(ITYPE(8),SL8,ZL)
      ENDIF
      SL1=2.1
      SL2=3.2
      SL3=4.3
      SL4=5.4
      SL5=6.5
      IF (ICOND.EQ.2.OR.ICOND.EQ.3) THEN
         ZL=ZL-0.3
         CALL MESSAG('PHL(1-10)  = $',100,SLT,ZL)
         CALL REALNO(PHL(1),105,SL1,ZL)
         CALL REALNO(PHL(2),105,SL2,ZL)
         CALL REALNO(PHL(3),105,SL3,ZL)
         CALL REALNO(PHL(4),105,SL4,ZL)
         CALL REALNO(PHL(5),105,SL5,ZL)
         ZL=ZL-0.3
         CALL REALNO(PHL(6),105,SL1,ZL)
         CALL REALNO(PHL(7),105,SL2,ZL)
         CALL REALNO(PHL(8),105,SL3,ZL)
         CALL REALNO(PHL(9),105,SL4,ZL)
         CALL REALNO(PHL(10),105,SL5,ZL)
      ENDIF
      IF (ICOND.EQ.4) THEN
         ZL=ZL-0.3
         CALL MESSAG('RABAMP(1-8)= $',100,SLT,ZL)
         CALL REALNO(RABAMP(1),105,SL1,ZL)
         CALL REALNO(RABAMP(2),105,SL2,ZL)
```

```
            CALL REALNO(RABAMP(3),105,SL3,ZL)
            CALL REALNO(RABAMP(4),105,SL4,ZL)
            CALL REALNO(RABAMP(5),105,SL5,ZL)
            ZL=ZL-0.3
            CALL REALNO(RABAMP(6),105,SL1,ZL)
            CALL REALNO(RABAMP(7),105,SL2,ZL)
            CALL REALNO(RABAMP(8),105,SL3,ZL)
            ZL=ZL-0.3
            CALL MESSAG('RDSLIM(1-8)= $',100,SLT,ZL)
            CALL REALNO(RDSLIM(1),105,SL1,ZL)
            CALL REALNO(RDSLIM(2),105,SL2,ZL)
            CALL REALNO(RDSLIM(3),105,SL3,ZL)
            CALL REALNO(RDSLIM(4),105,SL4,ZL)
            CALL REALNO(RDSLIM(5),105,SL5,ZL)
            ZL=ZL-0.3
            CALL REALNO(RDSLIM(6),105,SL1,ZL)
            CALL REALNO(RDSLIM(7),105,SL2,ZL)
            CALL REALNO(RDSLIM(8),105,SL3,ZL)
         ENDIF
         ZL=ZL-0.3
         CALL MESSAG('RINT(1-10) =  $',100,SLT,ZL)
         IPLACE=4
         IF (ABS(RINT(1)).GT.9999.0.OR.ABS(RINT(1)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(1),IPLACE,SL1,ZL)
         IPLACE=4
         IF (ABS(RINT(2)).GT.9999.0.OR.ABS(RINT(2)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(2),IPLACE,SL2,ZL)
         IPLACE=4
         IF (ABS(RINT(3)).GT.9999.0.OR.ABS(RINT(3)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(3),IPLACE,SL3,ZL)
         IPLACE=4
         IF (ABS(RINT(4)).GT.9999.0.OR.ABS(RINT(4)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(4),IPLACE,SL4,ZL)
         IPLACE=4
         IF (ABS(RINT(5)).GT.9999.0.OR.ABS(RINT(5)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(5),IPLACE,SL5,ZL)
         ZL=ZL-0.3
         IPLACE=4
         IF (ABS(RINT(6)).GT.9999.0.OR.ABS(RINT(6)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(6),IPLACE,SL1,ZL)
         IPLACE=4
         IF (ABS(RINT(7)).GT.9999.0.OR.ABS(RINT(7)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(7),IPLACE,SL2,ZL)
         IPLACE=4
         IF (ABS(RINT(8)).GT.9999.0.OR.ABS(RINT(8)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(8),IPLACE,SL3,ZL)
         IPLACE=4
         IF (ABS(RINT(9)).GT.9999.0.OR.ABS(RINT(9)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(9),IPLACE,SL4,ZL)
         IPLACE=4
         IF (ABS(RINT(10)).GT.9999.0.OR.ABS(RINT(10)).LT.0.01) IPLACE=-2
         CALL REALNO(RINT(10),IPLACE,SL5,ZL)
         IF (ICOND.EQ.3) THEN
            ZL=ZL-0.3
            CALL MESSAG('RTYPE      = $',100,SLT,ZL)
            CALL REALNO(RTYPE(1),105,SL1,ZL)
            CALL REALNO(RTYPE(2),105,SL2,ZL)
            CALL REALNO(RTYPE(3),105,SL3,ZL)
            CALL REALNO(RTYPE(4),105,SL4,ZL)
            CALL REALNO(RTYPE(5),105,SL5,ZL)
            ZL=ZL-0.3
            CALL REALNO(RTYPE(6),105,SL1,ZL)
            CALL REALNO(RTYPE(7),105,SL2,ZL)
```

```
          CALL REALNO(RTYPE(8),105,SL3,ZL)
       ENDIF
       IF (NT.GT.8) THEN
          ZL=ZL-0.3
          CALL MESSAG('TM(1,2)     = $',100,SLT,ZL)
          CALL REALNO(TM1,105,SL1,ZL)
          CALL REALNO(TM2,105,SL2,ZL)
          ZL=ZL-0.3
          CALL MESSAG('TOFF(1-10) =  $',100,SLT,ZL)
          CALL REALNO(TOFF(1),105,SL1,ZL)
          CALL REALNO(TOFF(2),105,SL2,ZL)
          CALL REALNO(TOFF(3),105,SL3,ZL)
          CALL REALNO(TOFF(4),105,SL4,ZL)
          CALL REALNO(TOFF(5),105,SL5,ZL)
          ZL=ZL-0.3
          CALL REALNO(TOFF(6),105,SL1,ZL)
          CALL REALNO(TOFF(7),105,SL2,ZL)
          CALL REALNO(TOFF(8),105,SL3,ZL)
          CALL REALNO(TOFF(9),105,SL4,ZL)
          CALL REALNO(TOFF(10),105,SL5,ZL)
          ZL=ZL-0.3
          CALL MESSAG('TWIDTH      = $',100,SLT,ZL)
          CALL REALNO(TWIDTH(1),105,SL1,ZL)
          CALL REALNO(TWIDTH(2),105,SL2,ZL)
          CALL REALNO(TWIDTH(3),105,SL3,ZL)
          CALL REALNO(TWIDTH(4),105,SL4,ZL)
          CALL REALNO(TWIDTH(5),105,SL5,ZL)
          ZL=ZL-0.3
          CALL REALNO(TWIDTH(6),105,SL1,ZL)
          CALL REALNO(TWIDTH(7),105,SL2,ZL)
          CALL REALNO(TWIDTH(8),105,SL3,ZL)
          CALL REALNO(TWIDTH(9),105,SL4,ZL)
          CALL REALNO(TWIDTH(10),105,SL5,ZL)
       ENDIF
       IF (NY.GT.8) THEN
          ZL=ZL-0.3
          CALL MESSAG('YOFF(1-10) =  $',100,SLT,ZL)
          CALL REALNO(YOFF(1),105,SL1,ZL)
          CALL REALNO(YOFF(2),105,SL2,ZL)
          CALL REALNO(YOFF(3),105,SL3,ZL)
          CALL REALNO(YOFF(4),105,SL4,ZL)
          CALL REALNO(YOFF(5),105,SL5,ZL)
          ZL=ZL-0.3
          CALL REALNO(YOFF(6),105,SL1,ZL)
          CALL REALNO(YOFF(7),105,SL2,ZL)
          CALL REALNO(YOFF(8),105,SL3,ZL)
          CALL REALNO(YOFF(9),105,SL4,ZL)
          CALL REALNO(YOFF(10),105,SL5,ZL)
          ZL=ZL-0.3
          CALL MESSAG('YM(1,2)     = $',100,SLT,ZL)
          CALL REALNO(YM1,105,SL1,ZL)
          CALL REALNO(YM2,105,SL2,ZL)
          ZL=ZL-0.3
          CALL MESSAG('YWIDTH      = $',100,SLT,ZL)
          CALL REALNO(YWIDTH(1),105,SL1,ZL)
          CALL REALNO(YWIDTH(2),105,SL2,ZL)
          CALL REALNO(YWIDTH(3),105,SL3,ZL)
          CALL REALNO(YWIDTH(4),105,SL4,ZL)
          CALL REALNO(YWIDTH(5),105,SL5,ZL)
          ZL=ZL-0.3
          CALL REALNO(YWIDTH(6),105,SL1,ZL)
          CALL REALNO(YWIDTH(7),105,SL2,ZL)
          CALL REALNO(YWIDTH(8),105,SL3,ZL)
```

```
            CALL REALNO(YWIDTH(9),105,SL4,ZL)
            CALL REALNO(YWIDTH(10),105,SL5,ZL)
         ENDIF
         CALL ENDPL(0)
  85     CONTINUE
         IF (LPRMT(3).EQ.0) GO TO 95
C
C - THIRD FRAME OF PARAMETERS
         CALL AREA2D(8.0,11.0)
         ZL=10.2
         CALL MESSAG('LIST OF INPUT PARAMETERS (CONTD)$',100,SLT,ZL)
         ZL=9.8
         CALL MESSAG('CSEC(1-19,1-8) = $',100,SLT,ZL)
         SL1=0.1
         SL2=1.0
         SL3=1.9
         SL4=2.8
         SL5=3.7
         SL6=4.6
         SL7=5.5
         SL8=6.4
         ZL=ZL-0.3
         CALL REALNO(REAL(CSEC(1,1)),104,SL1,ZL)
         CALL REALNO(AIMAG(CSEC(1,1)),104,SL2,ZL)
         CALL REALNO(REAL(CSEC(1,2)),104,SL3,ZL)
         CALL REALNO(AIMAG(CSEC(1,2)),104,SL4,ZL)
         CALL REALNO(REAL(CSEC(1,3)),104,SL5,ZL)
         CALL REALNO(AIMAG(CSEC(1,3)),104,SL6,ZL)
         CALL REALNO(REAL(CSEC(1,4)),104,SL7,ZL)
         CALL REALNO(AIMAG(CSEC(1,4)),104,SL8,ZL)
         ZL=ZL-0.2
         CALL REALNO(REAL(CSEC(1,5)),104,SL1,ZL)
         CALL REALNO(AIMAG(CSEC(1,5)),104,SL2,ZL)
         CALL REALNO(REAL(CSEC(1,6)),104,SL3,ZL)
         CALL REALNO(AIMAG(CSEC(1,6)),104,SL4,ZL)
         CALL REALNO(REAL(CSEC(1,7)),104,SL5,ZL)
         CALL REALNO(AIMAG(CSEC(1,7)),104,SL6,ZL)
         CALL REALNO(REAL(CSEC(1,8)),104,SL7,ZL)
         CALL REALNO(AIMAG(CSEC(1,8)),104,SL8,ZL)
         ZL=ZL-0.3
         CALL REALNO(REAL(CSEC(2,1)),104,SL1,ZL)
         CALL REALNO(AIMAG(CSEC(2,1)),104,SL2,ZL)
         CALL REALNO(REAL(CSEC(2,2)),104,SL3,ZL)
         CALL REALNO(AIMAG(CSEC(2,2)),104,SL4,ZL)
         CALL REALNO(REAL(CSEC(2,3)),104,SL5,ZL)
         CALL REALNO(AIMAG(CSEC(2,3)),104,SL6,ZL)
         CALL REALNO(REAL(CSEC(2,4)),104,SL7,ZL)
         CALL REALNO(AIMAG(CSEC(2,4)),104,SL8,ZL)
         ZL=ZL-0.2
         CALL REALNO(REAL(CSEC(2,5)),104,SL1,ZL)
         CALL REALNO(AIMAG(CSEC(2,5)),104,SL2,ZL)
         CALL REALNO(REAL(CSEC(2,6)),104,SL3,ZL)
         CALL REALNO(AIMAG(CSEC(2,6)),104,SL4,ZL)
         CALL REALNO(REAL(CSEC(2,7)),104,SL5,ZL)
         CALL REALNO(AIMAG(CSEC(2,7)),104,SL6,ZL)
         CALL REALNO(REAL(CSEC(2,8)),104,SL7,ZL)
         CALL REALNO(AIMAG(CSEC(2,8)),104,SL8,ZL)
         ZL=ZL-0.3
         CALL REALNO(REAL(CSEC(3,1)),104,SL1,ZL)
         CALL REALNO(AIMAG(CSEC(3,1)),104,SL2,ZL)
         CALL REALNO(REAL(CSEC(3,2)),104,SL3,ZL)
         CALL REALNO(AIMAG(CSEC(3,2)),104,SL4,ZL)
         CALL REALNO(REAL(CSEC(3,3)),104,SL5,ZL)
```

```
      CALL REALNO(AIMAG(CSEC(3,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(3,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(3,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(3,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(3,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(3,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(3,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(3,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(3,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(3,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(3,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(4,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(4,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(4,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(4,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(4,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(4,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(4,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(4,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(4,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(4,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(4,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(4,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(4,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(4,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(4,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(4,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(5,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(5,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(5,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(5,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(5,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(5,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(5,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(5,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(5,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(5,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(5,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(5,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(5,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(5,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(5,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(5,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(6,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(6,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(6,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(6,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(6,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(6,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(6,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(6,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(6,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(6,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(6,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(6,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(6,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(6,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(6,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(6,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(7,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(7,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(7,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(7,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(7,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(7,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(7,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(7,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(7,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(7,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(7,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(7,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(7,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(7,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(7,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(7,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(8,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(8,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(8,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(8,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(8,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(8,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(8,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(8,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(8,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(8,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(8,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(8,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(8,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(8,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(8,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(8,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(9,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(9,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(9,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(9,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(9,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(9,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(9,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(9,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(9,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(9,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(9,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(9,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(9,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(9,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(9,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(9,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(10,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(10,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(10,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(10,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(10,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(10,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(10,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(10,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(10,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(10,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(10,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(10,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(10,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(10,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(10,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(10,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(11,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(11,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(11,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(11,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(11,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(11,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(11,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(11,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(11,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(11,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(11,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(11,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(11,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(11,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(11,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(11,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(12,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(12,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(12,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(12,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(12,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(12,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(12,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(12,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(12,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(12,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(12,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(12,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(12,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(12,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(12,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(12,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(13,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(13,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(13,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(13,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(13,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(13,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(13,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(13,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(13,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(13,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(13,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(13,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(13,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(13,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(13,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(13,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(14,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(14,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(14,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(14,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(14,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(14,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(14,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(14,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(14,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(14,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(14,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(14,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(14,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(14,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(14,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(14,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(15,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(15,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(15,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(15,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(15,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(15,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(15,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(15,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(15,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(15,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(15,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(15,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(15,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(15,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(15,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(15,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(16,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(16,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(16,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(16,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(16,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(16,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(16,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(16,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(16,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(16,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(16,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(16,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(16,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(16,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(16,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(16,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(17,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(17,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(17,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(17,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(17,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(17,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(17,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(17,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(17,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(17,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(17,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(17,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(17,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(17,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(17,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(17,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(18,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(18,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(18,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(18,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(18,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(18,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(18,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(18,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(18,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(18,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(18,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(18,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(18,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(18,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(18,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(18,8)),104,SL8,ZL)
      ZL=ZL-0.3
      CALL REALNO(REAL(CSEC(19,1)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(19,1)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(19,2)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(19,2)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(19,3)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(19,3)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(19,4)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(19,4)),104,SL8,ZL)
      ZL=ZL-0.2
      CALL REALNO(REAL(CSEC(19,5)),104,SL1,ZL)
      CALL REALNO(AIMAG(CSEC(19,5)),104,SL2,ZL)
      CALL REALNO(REAL(CSEC(19,6)),104,SL3,ZL)
      CALL REALNO(AIMAG(CSEC(19,6)),104,SL4,ZL)
      CALL REALNO(REAL(CSEC(19,7)),104,SL5,ZL)
      CALL REALNO(AIMAG(CSEC(19,7)),104,SL6,ZL)
      CALL REALNO(REAL(CSEC(19,8)),104,SL7,ZL)
      CALL REALNO(AIMAG(CSEC(19,8)),104,SL8,ZL)
      CALL RESET('HEIGHT')
      CALL ENDPL(0)
 95   CONTINUE
C
C -- PRECALCULATE 'NICE' END VALUES AND INTERVALS FOR FREQUENTLY USED
C    COORDINATE AXES, NUMERICAL STEP SIZE
C
C - T-AXIS
      IF (NT.GT.8) THEN
         TORIG=TM1
         TSTP=2.0
         TMAX=TM2
         NECLEC=-1
         CALL NYSXIS(TORIG,1,NECLEC,TORIG,TSTP,TMAX)
         RDT=(TM2-TM1)/NT
      ENDIF
      IF (NY.GT.0) THEN
C
C - Y-AXIS
         RDY=(YM2M1)/NY
         YORIG=YM1
         YSTP=2.0
         YMAX=YM2
         NECLEC=-1
         CALL NYSXIS(YORIG,1,NECLEC,YORIG,YSTP,YMAX)
C
C - FFT-AXIS
         YFMAX=0.5/RDY
         YFORIG=-YFMAX
         YFSTP=YFMAX-YFORIG
         RDYF=YFSTP/NY
         WFORIG=YFORIG
         WFSTP=0.0
         WFMAX=YFMAX
         NECLEC=-1
         CALL NYSXIS(WFORIG,1,NECLEC,WFORIG,WFSTP,WFMAX)
         WFSTP=2.0
         NECLEC=-1
         CALL NYSXIS(WFORIG,1,NECLEC,WFORIG,WFSTP,WFMAX)
         WFORIG=WFORIG+WFSTP
         WFMAX=WFMAX-WFSTP
      ENDIF
C
C - INTENSITY NORMALIZATION FACTOR
      R1=8.0*PI/SPEED
      R2=R1*YM2M1*YM2M1
C
C --- LOOP THROUGH: ALL RECORDS IN DATA FILE / ALL DATA FILES
C
      IF (NT.GT.0.AND.NY.GT.0) THEN
         IF (DONYET.EQ.0) THEN
            NWRT=4
            NI0=2
         ELSE
            NI0=(NWRT-2)/6
         ENDIF
      ELSE
         IF (NY.GT.0) THEN
            NI0=(NWRT-2)/6
         ELSE
            NI0=(NWRT-2)/3
         ENDIF
      ENDIF
      WRITE (59,*)'NI0= ',NI0
      IZ=1
      KZOLD=0
      KZNEW=KZ(1)
C
      DO 500 I0=1,NI0
      WRITE (59,*)'I0= ',I0
      WRITE (59,*)'KZOLD= ',KZOLD
      WRITE (59,*)'KZNEW= ',KZNEW
      WRITE (59,*)'IZ= ',IZ
      IF (KZNEW.LE.KZOLD.AND.I0.GT.2) THEN
         WRITE (59,*)'KZNEW .LE. KZOLD; STOP AT I0= ',I0
         GOTO 501
      ENDIF
      IUNIT=4
      IF (NT.GT.8.AND.NY.GT.8.AND.KZNEW.EQ.1) IUNIT=3
      IF (I0.NE.KZNEW) THEN
C
C - NO PLOTTING AT CURRENT ZVAL, MOVE ON IN DATA FILE/S
         IF (NT.GT.8.AND.NY.GT.8) THEN
            IF (I0.GT.1) THEN
               ISPCT=ISPCT+1
            ENDIF
            GO TO 500
         ENDIF
         GO TO 290
      ENDIF
      IF (I0.NE.1) THEN
         IF (NT.GT.8.AND.NY.GT.8) THEN
            IF (DONYET.NE.0) THEN
               ISPCT=ISPCT+1
            ELSE
               ISPCT=IPART
            ENDIF
            INUMRL=INT(ISPCT/100)
            ISTP1=NUMRAL (INUMRL+1:INUMRL+1)
            IRST=ISPCT-100*INUMRL
            INUMRL=INT(IRST/10)
            ISTP2=NUMRAL (INUMRL+1:INUMRL+1)
            INUMRL=IRST-10*INUMRL
            ISTP3=NUMRAL (INUMRL+1:INUMRL+1)
            DTFL2D=FRM//'2'
            PDN2D=FRAM//ISTP1//ISTP2//ISTP3
            WRITE (59,*)'=DTFL2D ',DTFL2D
            WRITE (59,*)'PDN2D= ',PDN2D
CDIR$ BLOCK
            CALL ACCESS(IRRE,'DN'L,DTFL2D,'PDN'L,PDN2D,'ED'L,EDN)
            CALL ASSIGN(IRRE,'DN'L,DTFL2D,'A'L,'FT04'L)
         ENDIF
      ENDIF
      IF (KZNEW.GT.1.OR.LPRMT(4).NE.1.OR.NY.LE.8) GO TO 175
C
C -- INITIALLY, WHEN NY.GT.8, COMPUTE DIVERSE WAVE NUMBER AVERAGES
C    AND WRITE THEM ONTO A SEPARATE GRAPHICS OUTPUT FRAME
      CALL AREA2D(8.0,10.5)
      SLT=2.0
      ZL=10.2
      CALL MESSAG('LIST OF OUTPUT PARAMETERS$',100,SLT,ZL)
      SLT=0.5
      ZL=ZL-0.7
      IF (NT.GT.8) THEN
C
C - READ INITIAL PUMP DATA, COMPUTE TOTAL PUMP ENERGY
         READ (IUNIT) ZVAL,AEQ
         WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
         TPI=0.0
         DO 105 I2=1,NY
         DO 105 I3=1,NT
         TPI=TPI+AEQ(I3,I2)*CONJG(AEQ(I3,I2))
 105     CONTINUE
         TPI=TPI*RDY*RDT/R1
         CALL MESSAG('COMBINED LINEAR ENERGY DENSITY OF PUMPS IN MILLIJ
     1OULE/CM = $',100,SLT,ZL)
         SLTR=SLT+3.0
         ZL=ZL-0.35
         IPLACE=2
         IF (ABS(TPI).GT.9999.9.OR.ABS(TPI).LT.0.01) IPLACE=-2
         CALL REALNO(TPI,IPLACE,SLTR,ZL)
         WRITE (59,*)'TOTAL LINEAR ENERGY DENSITY IN PUMPS IN ',
     1   'MILLIJOULES/CM = ',TPI
C
C - READ INITIAL STOKES DATA, COMPUTE TOTAL STOKES ENERGY
         READ (IUNIT) ZVAL,AEQ
         WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
         TPI=0.0
         DO 107 I2=1,NY
         DO 107 I3=1,NT
         TPI=TPI+AEQ(I3,I2)*CONJG(AEQ(I3,I2))
 107     CONTINUE
         TPI=TPI*RDY*RDT/R1
         ZL=ZL-0.5
         CALL MESSAG('COMBINED LINEAR ENERGY DENSITY OF STOKES IN MILLI
     1JOULE/CM = $',100,SLT,ZL)
         IPLACE=2
         IF (ABS(TPI).GT.9999.9.OR.ABS(TPI).LT.0.01) IPLACE=-2
         ZL=ZL-0.35
         CALL REALNO(TPI,IPLACE,SLTR,ZL)
         WRITE (59,*)'TOTAL LINEAR ENERGY DENSITY IN STOKES IN ',
     1   'MILLIJOULES/CM = ',TPI
         CALL SKIPR(IUNIT,1,ISTAT)
         WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1   ' FILES IN UNIT ',IUNIT,' .'
      ELSE
         CALL SKIPR(IUNIT,3,ISTAT)
         WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1   ' FILES IN UNIT ',IUNIT,' .'
      ENDIF
C
C - INITIAL PUMP BEAMS FFT INTENSITY DATA, READ PUMP FFT DATA AND
C   RESET FILE POINTER
      READ (IUNIT) ZVAL,AEQ
      WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
      CALL SKIPR(IUNIT,-4,ISTAT)
      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),'
     1 FILES IN UNIT ',IUNIT,' .'
      DO 110 I2=1,NY
      DO 110 I3=1,NT
      SRFI(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/R2
 110  CONTINUE
C
C - DETERMINE INITIAL SEQUENCE OF INDICES OF PUMP BEAMS ALONG Y-AXIS
C   FROM LEFT TO RIGHT
      IF (NPUMP.EQ.1) INDEX(1)=1
      IF (NPUMP.EQ.2) THEN
         IF (YOFF(1).LT.YOFF(2)) THEN
            INDEX(1)=1
            INDEX(2)=2
         ELSE
            INDEX(1)=2
            INDEX(2)=1
         ENDIF
      ELSE IF (NPUMP.GT.2) THEN
         MODE=2
         CALL ORDERS(MODE,IWORK,SRTYOF,INDEX,NPUMP,1,8,1)
      ENDIF
      WRITE (59,*) 'PUMP INDICES SEQUENTIALLY',INDEX
C
C - ERROR CONDITIONS AND WARNINGS
      IF (YM(1).GT.YOFF(INDEX(1))) THEN
         WRITE (59,*) 'FIRST BEAM OUTSIDE Y-WINDOW; STOP'
          CALL EXIT(1)
      ENDIF
      IF (YM(2).LT.YOFF(INDEX(NPUMP))) THEN
         WRITE (59,*) 'LAST BEAM OUTSIDE Y-WINDOW; STOP'
          CALL EXIT(1)
      ENDIF
      IF (NPUMP.GT.1) THEN
         DO 120 INP=1,NPUMP-1
         IB=INDEX(INP)
         IBNX=INDEX(INP+1)
         IF (YOFF(IBNX)-YOFF(IB).LE.YWIDTH(IBNX)+YWIDTH(IB)) THEN
            WRITE (59,*) 'BEAM ',IB,' AND BEAM ',IBNX,
     1                     ' ARE TOO CLOSE FOR AVERAGE K CALCULATIONS'
         ENDIF
         RIB=RINT(IBNX)/RINT(IB)
         IF (RIB.LT.0.1.OR.RIB.GT.10.0) THEN
            WRITE (59,*) 'DISPARATE INTENSITIES OF BEAM ',IB,' AND ',
     1              IBNX,'MAY OBSCURE SIGNATURES IN AVRG. K CALCS.'
         ENDIF
 120     CONTINUE
      ENDIF
C
C - CONSIDER EACH BEAM AT TIME TOFF WHEN 2-D; CONSIDER ALL NT CASES
C   WHEN 1-D (STATIONARY)
      SL=SL+0.5
      ZL=ZL-0.5
      CALL MESSAG('PUMP$',100,SLT,ZL)
      SLT=SLT+1.2
      CALL MESSAG('TOTAL INTENSITY$',100,SLT,ZL)
      SLT=SLT+2.6
      CALL MESSAG('K-WIDTH$',100,SLT,ZL)
      IFRM=0
      NCASES=NT
      IF (NT.GT.8) NCASES=1
      DO 170 ICS=1,NCASES
      IF (NCASES.GT.1) THEN
         IF (ZL.LT.NPUMP*0.3+0.5) THEN
            IFRM=IFRM+1
            CALL ENDPL(0)
            CALL AREA2D(8.0,10.5)
            ZL=10.2
            CALL MESSAG('LIST OF OUTPUT PARAMETERS (CONTD)$',
     1                  100,SLT,ZL)
            ZL=9.5
            CALL MESSAG('PUMP$',100,SLT,ZL)
            SLT=SLT+1.2
            CALL MESSAG('TOTAL INTENSITY$',100,SLT,ZL)
            SLT=SLT+2.6
            CALL MESSAG('K-WIDTH$',100,SLT,ZL)
            IF (IFRM.GT.8) THEN
               WRITE (59,*) 'LIST OF OUTPUT PARAMETER INTERUPTED'
               GO TO 170
            ENDIF
         ENDIF
         SLT=0.1
         ZL=ZL-0.4
         CALL MESSAG('CASE$',100,SLT,ZL)
         SLT=SLT+0.8
         CALL INTNO(ICS,SLT,ZL)
      ENDIF
      IT=ICS
      ZL=ZL-0.1
C
C - CONSIDER ONE BEAM AFTER THE OTHER (ALONG K-AXIS FROM RIGHT TO LEFT)
```

```
      DO 160 INP=1,NPUMP
      IB=INDEX(INP)
C
C - RIGHT BEAM LIMIT
      IF (INP.EQ.1) THEN
         JR=NY
      ELSE
         JR=JL
      ENDIF
C
C - LEFT BEAM LIMIT
      IF (INP.EQ.NPUMP) THEN
         JL=1
      ELSE
         IBNX=INDEX(INP+1)
         YKL=-0.5*(YOFF(IB)+YOFF(IBNX))*RKP/ZINT
         JL=ANINT((YKL-YFORIG)*YM2M1)
      ENDIF
C
C - SELECT TEMPORAL PEAK OF EACH BEAM IN 2-D CASES
      IF (NT.GT.8) THEN
         TOFIB=TOFF(IB)
         IF (TOFIB.LT.TM1.OR.TOFIB.GT.TM2) THEN
            WRITE (59,*) 'BEAM ',IB,' IS OUTSIDE T-RANGE'
            GO TO 160
         ENDIF
         IT=ANINT(NT*(TOFIB-TM1)/(TM2-TM1))
      ENDIF
C
C - COMPUTE TOTAL INTENSITY OF BEAM IN K-SPACE
      TIK(JL)=0.0
      DO 140 J=JL+1,JR
      TIK(J)=TIK(J-1)+SRFI(IT,J)*RDYF
  140 CONTINUE
      WRITE (59,*)'TIK = ',TIK
      TIKBMX=TIK(JR)
      WRITE (59,*)'JL= ',JL,' JR= ',JR,' TIKBMX= ',TIKBMX,' IT= ',IT
      ZL=ZL-0.3
      SLT=0.7
      CALL INTNO(IB,SLT,ZL)
      SLT=SLT+1.3
      IPLACE=2
      IF (ABS(TIKBMX).GT.9999.0.OR.ABS(TIKBMX).LT.0.01) IPLACE=-2
      CALL REALNO(TIKBMX,IPLACE,SLT,ZL)
      WRITE (59,*)'TOTAL INTENSITY IN K= ',TIKBMX
C
C - COMPUTE K-WIDTH OF BEAM (LINEAR INTERPOLATION)
      IF (TIKBMX.GT.1.0E-64) THEN
         DO 150 J=JL,JR
         TIK(J)=ABS((2.0*TIK(J)-TIKBMX)/TIKBMX)-WDLIM
  150    CONTINUE
      ELSE
         WRITE (59,*)'POWER INTERGRAL IN K-SPACE VANISHES'
      ENDIF
      JSRCH=JR-JL+1
      CALL WHENFLE(JSRCH,TIK(JL),1,0.0,IWHEN,NVAL)
      WRITE (59,*)'IWHEN = ',IWHEN
      IWN1=IWHEN(1)+JL-1
      IWN2=IWHEN(NVAL)+JL-1
      IF (NPUMP.GT.1.AND.(IWN1.EQ.1.OR.IWN2.EQ.NY)) THEN
         WRITE (59,*) 'AVERAGE K CALCULATION FAILED'
         GOTO 170
      ENDIF
      IF (IWN2-IWN1.LT.4) THEN
         WRITE (59,*) 'INSUFFICIENT RESOLUTION FFT BEAM ',IB
         GOTO 170
      ENDIF
      TIKDL1=TIK(IWN1-1)-TIK(IWN1)
      TIKDL2=TIK(IWN2+1)-TIK(IWN2)
      WRITE (59,*)'TIKDL1= ',TIKDL1,' TIKDL2= ',TIKDL2
      YKL=YFORIG+RDYF*(IWN1+TIK(IWN1)/(TIK(IWN1-1)-TIK(IWN1)))
      YKR=YFORIG+RDYF*(IWN2-TIK(IWN2)/(TIK(IWN2+1)-TIK(IWN2)))
      WDTHKB=YKR-YKL
      SLT=SLT+2.4
      IPLACE=2
      IF (ABS(WDTHKB).GT.9999.0.OR.ABS(WDTHKB).LT.0.01) IPLACE=-2
      CALL REALNO(WDTHKB,IPLACE,SLT,ZL)
  160 CONTINUE
  170 CONTINUE
      CALL ENDPL(0)
  175 CONTINUE
C
C --- GENERATE DESIRED GRAPHICS DATA AND CALL PLOTTING SUBROUTINES
C
      IPLTGP=1
      IF (NY.LE.8) IPLTGP=2
      DO 220 IPLT=1,6,IPLTGP
C
C - CHECK WHICH GRAPHS ARE REQUESTED
      IF (NY.GT.8.AND.NT.GT.8) THEN
         JSRF=ISRF(IPLT)
      ELSE
         JSRF=0
      ENDIF
      SCI=0.0
      LCSEC=3*(IPLT-1)+1
      DO 180 IS=1,NSEC
      SCI=SCI+ABS(AIMAG(CSEC(LCSEC,IS)))
  180 CONTINUE
      SCA=0.0
      LCSEC=LCSEC+1
      DO 190 IS=1,2*NSEC
      IFLIP=INT((IS-1)/NSEC)
      NSC=LCSEC+IFLIP
      ISS=IS-IFLIP*NSEC
      SCA=SCA+ABS(AIMAG(CSEC(NSC,ISS)))
  190 CONTINUE
C
C - WHEN A GRAPH IS REQUESTED READ ITS AMPLITUDE DATA IN AND RESET
C   FILE POINTER TO FIRST OF THE RECORDS WITH THE CURRENT ZVAL
      WRITE (59,*)'SCI,SCA,I0,IPLT,LCSEC,JSRF,IZ,KZNEW,NWRT'
      WRITE (59,*) SCI,SCA,I0,IPLT,LCSEC,JSRF,IZ,KZNEW,NWRT
      IF (SCI.GT.0.001.OR.SCA.GT.0.001.OR.JSRF.NE.0) THEN
C
C - MATCH UP EL,ES,Q,AEL,AES,AQ STORAGE WITH EL,AEL,ES,AES,Q,AQ
C   PLOTTING
         IF (IPLT.EQ.1) IRCD=1
         IF (IPLT.EQ.2) IRCD=4
         IF (IPLT.EQ.3) IRCD=2
         IF (IPLT.EQ.4) IRCD=5
         IF (IPLT.EQ.5) IRCD=3
         IF (IPLT.EQ.6) IRCD=6
C        IRCD=1+MOD(IPLT+2*(IPLT-1)+INT(IPLT/2)-1,6)
         CALL SKIPR(IUNIT,IRCD-1,ISTAT)
         WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1   ' FILES IN UNIT ',IUNIT
```

```
          READ (IUNIT) ZVAL,AEQ
          WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
          CALL SKIPR(IUNIT,-IRCD,ISTAT)
      WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1  ' FILES IN UNIT ',IUNIT,' .'
      ELSE
          GO TO 220
      ENDIF
C - INTENSITY CONTOURS
      IF (JSRF.NE.0) THEN
          DO 200 I2=1,NY
          DO 200 I3=1,NT
          SRF(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/R1
  200     CONTINUE
          CALL CNTR(IPLT*JSRF)
      ENDIF
C
C - INTENSITY SECTIONS
      IF (SCI.GT.0.001) THEN
          X1=R1
          IF (IPLT.EQ.2.OR.IPLT.EQ.4.OR.IPLT.EQ.6) X1=R2
          DO 205 I2=1,NY
          DO 205 I3=1,NT
          SRF(I3,I2)=AEQ(I3,I2)*CONJG(AEQ(I3,I2))/X1
  205     CONTINUE
          CALL CRSSCT(LCSEC-1)
      ENDIF
C
C - LOAD PHASE AND AMPLITUDE DATA
      IF (SCA.GT.0.001) THEN
          DO 210 I2=1,NY
          DO 210 I3=1,NT
          SRF(I3,I2)=REAL(AEQ(I3,I2))
          SRFI(I3,I2)=AIMAG(AEQ(I3,I2))
  210     CONTINUE
C
C - PHASE AND AMPLITUDE SECTIONS
          CALL CRSSCT(LCSEC)
      ENDIF
  220 CONTINUE
C
C -- PLOTS WITH SUM OF THE INTENSITIES OF PUMP BEAMS AND STOKES BEAM
C    AND LONGITUDINAL INVARIANTS
C
      SC=0.0
      DO 230 IS=1,NSEC
      SC=SC+ABS(AIMAG(CSEC(19,IS)))
  230 CONTINUE
C
C - DATA OF PUMPS AND STOKES INTENSITY COMBINED
      IFLG=0
      IF (ISRF(7).NE.0.AND.NT.GT.8.AND.NY.GT.8) THEN
          JSRF=ISRF(7)
          READ (IUNIT) ZVAL,AEQ
          READ (IUNIT) ZVAL,AER
          WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
          WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
          IFLG=1
          CALL SKIPR(IUNIT,-2,ISTAT)
          WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
     1    ISTAT(2),' FILES IN UNIT ',IUNIT,' .'
          DO 250 I2=1,NY
          DO 250 I3=1,NT
```

```
         SRF(I3,I2)=(AEQ(I3,I2)*CONJG(AEQ(I3,I2))
   1                +AER(I3,I2)*CONJG(AER(I3,I2)) )/R1
 250     CONTINUE
         CALL CNTR(7*JSRF)
      ENDIF
C
C - DATA OF INVARIANT ALONG Z
      IF (SC.GT.0.001) THEN
C
C - LONGITUDINAL INVARIANT IN 1-D TRANSIENT CASE
         IF (IFLG.EQ.0) THEN
            READ (IUNIT) ZVAL,AEQ
            READ (IUNIT) ZVAL,AER
            WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
            WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
            CALL SKIPR(IUNIT,-2,ISTAT)
            WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
     1      ISTAT(2),' FILES IN UNIT ',IUNIT,' .'
         ENDIF
         DO 255 I2=1,NY
         DO 255 I3=1,NT
         SRF(I3,I2)=(RKS*AEQ(I3,I2)*CONJG(AEQ(I3,I2))
     1              +RKP*AER(I3,I2)*CONJG(AER(I3,I2)) )/R1
 255     CONTINUE
         IF (NY.GT.8) THEN
C
C - LONGITUDINAL INVARIANT IN 1-D STATIONARY CASE AND IN 2-D
            DO 257 I3=1,NT
            SRF(I3,1)=0.0
 257        CONTINUE
            DO 258 I2=2,NY
            DO 258 I3=1,NT
            SRF(I3,I2)=SRF(I3,I2)*RDY+SRF(I3,I2-1)
 258        CONTINUE
         ENDIF
         CALL CRSSCT(19)
      ENDIF
      IF (ISRF(8).NE.0.AND.NT.GT.8.AND.NY.GT.8) THEN
         JSRF=ISRF(8)

C
C - DATA OF PUMPS AND STOKES FFT INTENSITY COMBINED
         CALL SKIPR(IUNIT,3,ISTAT)
         WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1                ' FILES IN UNIT ',IUNIT
         READ (IUNIT) ZVAL,AEQ
         READ (IUNIT) ZVAL,AER
         WRITE (59,*) 'READ (IUNIT) ZVAL,AEQ'
         WRITE (59,*) 'READ (IUNIT) ZVAL,AER'
         CALL SKIPR(IUNIT,-5,ISTAT)
         WRITE (59,*) 'SKIPPED BACK ',ISTAT(1),' RECORDS AND ',
     1   ISTAT(2),' FILES IN UNIT ',IUNIT,' .'
         DO 260 I2=1,NY
         DO 260 I3=1,NT
         SRF(I3,I2)=( AEQ(I3,I2)*CONJG(AEQ(I3,I2))
   1                +AER(I3,I2)*CONJG(AER(I3,I2)) )/R2
 260     CONTINUE
         CALL CNTR(8*JSRF)
      ENDIF
C
C ---- END OF PLOTTING DATA SET
C
      IF (NT.GT.8.AND.NY.GT.8.AND.I0.GT.1) THEN
```

```
            CALL RELEASE(IRRE,'DN'L,DTFL2D)
            WRITE (59,*)'RELEASED DN= ',DTFL2D
         ENDIF
         KZOLD=KZNEW
         IZ=IZ+1
         KZNEW=KZ(IZ)
  290  CONTINUE
C
C -- MOVE POINTER IN DATA FILE ON TO FIRST RECORD WITH NEXT ZVAL
C
         IF (NY.GT.8) THEN
            CALL SKIPR(IUNIT,6,ISTAT)
            WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1                ' FILES IN UNIT ',IUNIT
         ELSE
            CALL SKIPR(IUNIT,3,ISTAT)
            WRITE (59,*) 'SKIPPED ',ISTAT(1),' RECORDS AND ',ISTAT(2),
     1                ' FILES IN UNIT ',IUNIT
         ENDIF
  500  CONTINUE
C
C --- CLOSE GRAPHICS SURFACES, END PROGRAM
C
  501  CONTINUE
       CALL DONEPL
       CALL EXIT(1)
       END
c
c
c
c
       SUBROUTINE CNTR(KSRF)
c
C  This subroutine was written by Godehard Hilfer and Curtis R. Menyuk
C  (2/87). It uses the commercial graphics package DISSPLA (SDSS) to
C  generate a contour plot of the data in array srf.
c
C  This subroutine employs the subroutine nyaxis to compute 'nice' tick
C  marks along the coordinate axes and the subroutine xisFFT to compute
C  the location, extremas and intervals of the transformed variable axis
C  in FFT-plots. Depending on the value of ksrf various titles,
C  coordinate axes and labels are selected and drawn.  The sign of
C  ksrf toggles the labeling option of the main contour lines
C  (positive ksrf labels, negative ksrf no labels). The main contour
C  lines are solid lines representing integral powers of 10. ndeC such
C  lines will be drawn below the surface maximum. lin (<9) other
C  contour lines (dashed lines) are drawn between the main contour
C  lines corresponding to the integral multiples of the next lower
C  integral power of ten. Which integral multiples are drawn is
C  determined by the first lin elements of the vector level. If
C  ishm = 1 a dotted contour will mark the half-height level, if
C  ishm = 0 this line will not be drawn, if ishm =-1 the half-height
C  contour and a dot at the surface maximum should be drawn.
C
C                         --variables--
c
C     grfsz = physical size of graphics plots
C     i2 = y-coordinate index in do-loops 228,230
C     i3 = t-coordinate index in do-loops 225,230
C     lin = number of dashed contours between solid contours
C     ishm = flag for half-height contour option in sub=cntr
C     ksrf = index number of surface that is being contoured
C     labl = labelling variable
```

```
C       level = vector with desired level heights for dashed contours
C       lv = index do-loop 240
C       ndeC = desired number of solid contours representing powers of 10
C       necveC = data switch for subroutine nyaxis
C       nt = see RAM2D1
C       ntp = nt+1
C       ny = see RAM2D1
C       nyp = ny+1
C       rdy = step size in transverse spatial variable y
C       srf = array of data from which contours are plotted
C       tm1 = time coordinate lower limit
C       tm2 = time coordinate upper limit
C       tmax = value at end of time axis
C       torig = value at beginning of time axis
C       tstp = time axis interval
C       wfmax = nice spatial FFT axis end value
C       wforig = nice spatial FFT axis beginning value
C       wfstp = nice spatial FFT axis interval
C       xdum = dummy variable holding the x-coordinate of two points
C       ydum = dummy variable holding the y-coordinate of two points
C       yfmax = value at end of spatial FFT axis
C       yforig = value at beginning of spatial FFT axis
C       yfstp = spatial FFT axis interval
C       ymax = value at end of transverse spatial axis
C       yorig = value at beginning of transverse spatial axis
C       ystp = transverse spatial axis interval
C       ym1 = y-coordinate lower limit
C       ym2 = y-coordinate upper limit
C       zbot = logarithmiC data cutoff
C       zincr = special contour separation
C       zlev = integral power of 10 next to data maximum
C       zmax = data maximum
C       zplane = reference level for contours
C       zval = value of z-coordinate of current data/plot
C ----------------------------------------------------------------------
c
        PARAMETER (NT=256,NTP=NT+1,NX=8,NY=128,NYP=NY+1,NYTP=NYP*NTP)
C
        IMPLICIT COMPLEX(A-E,Q)
        DIMENSION ISRF(8),ITYPE(8),LEVEL(8),CSEC(19,NX),RTYPE(8),
       1          SRF(NTP,NYP),SRFI(NTP,NYP),SRFSEQ(NYTP),XDUM(2),YDUM(2)
        COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
       1        GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
       2        YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
       3        ZVAL
        COMMON /NUM/ RDT,RDY,RDYF,TM1,TM2,YM1,YM2,YM2M1
        COMMON WORK(25000)
        EQUIVALENCE (SRF,SRFSEQ)
C
C - NO CONTOURING IN ONE DIMENSIONAL CASES
        IF (NT.LE.8.OR.NY.LE.8) RETURN
C
C - TOGGLE LABELLING DEPENDING ON SIGN OF KSRF
        LABL='LABELS'
        IF (KSRF.LT.0) LABL='NOLABELS'
        KSRF=ABS(KSRF)
C
C - SURFACE DATA
C - MAKE DATA ARRAY SYMMETRIC TO OBTAIN AXIS LABELS ('NICE') AT AXIS END
        DO 200 I3=1,NT
        SRF(I3,NYP)=SRF(I3,1)
  200   CONTINUE
        DO 210 I2=1,NY
```

```
      SRF(NTP,I2)=SRF(1,I2)
 210  CONTINUE
C
C - FIND DATA MAXIMUM, ITS INTEGRAL POWER OF TEN AND CORRESPONDING
C   MANTISSA
      ZMAX=SRFSEQ(ISMAX(NYTP-1,SRFSEQ,1))
      IF(ZMAX.LE.0.0)THEN
         WRITE (59,*) 'warning: ZMAX IS ZERO OR NEGATIVE WHEN KSRF =',
     1                KSRF
         RETURN
      ENDIF
C
C - DETERMINE LOWER DATA CUTOFF
      ZLEV=10.0**INT(ALOG10(ZMAX))
      ZBOT=(INT(ZMAX/ZLEV)+0.5)*ZLEV/10.0**NDEC
      IF(ZBOT.LE.0.0) WRITE (59,*) 'warning: ZBOT IS ZERO OR NEGATIVE ',
     1                              ' WHEN KSRF = ',KSRF
      SRF(NTP,NYP)=ZBOT
C
C - START A NEW GRAPHICS FRAME FOR THIS CONTOUR PLOT
      CALL RESET('ALL')
      CALL INTAXS
      CALL AREA2D(GRFSZ,GRFSZ)
C
C - HEADLINE, LABELS, AND COORDINATE SYSTEM
      GO TO (211,212,213,214,215,216,217,218) KSRF
211   CALL HEADIN('TRANSIENT RAMAN: PUMP (PWR)$',100,1.5,1)
      GO TO 222
212   CALL HEADIN('TRANSIENT RAMAN: PUMP (FFT, PWR)$',100,1.5,1)
      GO TO 222
213   CALL HEADIN('TRANSIENT RAMAN: STOKES (PWR)$',100,1.5,1)
      GO TO 222
214   CALL HEADIN('TRANSIENT RAMAN: STOKES (FFT, PWR)$',100,1.5,1)
      GO TO 222
215   CALL HEADIN('TRANSIENT RAMAN: MATERIAL EXCITATION$',100,1.5,1)
      GO TO 222
216   CALL HEADIN('TRANSIENT RAMAN: MATERIAL EXCITATION (FFT)$',
     1   100,1.5,1)
      GO TO 222
217   CALL HEADIN('TRANSIENT RAMAN: PUMP AND STOKES (PWR)$',100,1.5,1)
      GO TO 222
218   CALL HEADIN('TRANSIENT RAMAN: PUMP AND STOKES (FFT, PWR)$',
     1   100,1.5,1)
222   CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      CALL XNAME('TIME (PICO-SECONDS)$',100)
      IF(KSRF.EQ.2.OR.KSRF.EQ.4.OR.KSRF.EQ.6.OR.KSRF.EQ.8) THEN
         CALL YNONUM
         CALL YTICKS(0)
         VORIG=YFORIG
         VSTP=YFSTP
         VMAX=YFMAX
      ELSE
         CALL YNAME ('Y-DIMENSION (CM)$',100)
         VORIG=YORIG
         VSTP=YSTP
         VMAX=YMAX
C
C - AXIS LINE AND TICKMARKS ON THE RIGHT IN NO-FFT PLOTS
         NTIK=NINT((VMAX-VORIG)/VSTP)
```

```
               XDUM(1)=TMAX-(TMAX-TORIG)/50.0
               XDUM(2)=TMAX
               YDM=VORIG
               DO 225 ITK=1,NTIK-1
               YDM=YDM+VSTP
               YDUM(1)=YDM
               YDUM(2)=YDM
               CALL CURVE(XDUM,YDUM,2,0)
 225           CONTINUE
            ENDIF
            CALL GRAF(TORIG,TSTP,TMAX,VORIG,VSTP,VMAX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
            NTIK=NINT((TMAX-TORIG)/TSTP)
            YDUM(1)=VMAX
            YDUM(2)=VMAX-(VMAX-VORIG)/50.0
            XDM=TMAX
            DO 226 ITK=1,NTIK-1
            XDM=XDM-TSTP
            XDUM(1)=XDM
            XDUM(2)=XDM
            CALL CURVE(XDUM,YDUM,2,0)
 226        CONTINUE
            XDUM(1)=TORIG
            XDUM(2)=TMAX
            YDUM(1)=VMAX
            YDUM(2)=VMAX
            CALL CURVE(XDUM,YDUM,2,0)
            XDUM(1)=TMAX
            XDUM(2)=TMAX
            YDUM(1)=VMAX
            YDUM(2)=VORIG
            CALL CURVE(XDUM,YDUM,2,0)
            XDUM(1)=TORIG
            XDUM(2)=TORIG
            YDUM(1)=VMAX
            YDUM(2)=VORIG
            CALL CURVE(XDUM,YDUM,2,0)
C
C - PREPARE CONTOUR FINDING
            CALL BCOMON(25000)
            CALL CONANG(90.0)
            CALL PSPLIN
C
C - DRAW HALF-HEIGHT CONTOUR WITH LINE TYPE SPECIFIED IN
C   SUBROUTINE MYCON
            IF(ISHM.EQ.1.OR.ISHM.EQ.-1) THEN
               ZPLANE=0.1*ZMAX
               ZINCR=8.0*ZPLANE
               IF (ISHM.EQ.-1) THEN
                  ZINCR=ZINCR*(1.0-1.0E-5)
               ELSE
                  ZINCR=ZINCR*(1.0+1.0E-5)
               ENDIF
               CALL ZBASE(ZPLANE)
               CALL CONMAK(SRF,NTP,NYP,ZINCR)
               CALL CONLIN(0,'MYCON','NOLABELS',1,5)
               CALL CONTUR(1,'NOLABELS','DRAW')
               ZPLANE=0.5*ZMAX
               ZINCR=ZPLANE
               IF (ISHM.EQ.-1) THEN
                  ZINCR=ZINCR*(1.0-1.0E-5)
               ELSE
            ZINCR=ZINCR*(1.0+1.0E-5)
         ENDIF
         CALL ZBASE(ZPLANE)
         CALL CONMAK(SRF,NTP,NYP,ZINCR)
         CALL CONLIN(0,'MYCON','NOLABELS',1,5)
         CALL CONTUR(1,'NOLABELS','DRAW')
      ENDIF
C
C - REPLACE ALL DATA SMALLER THAN THE CUTOFF ZBOT BY ZBOT
C   TO ELIMINATE UNDESIRED LOGARITHMIC CONTOURS
      DO 230 I2=1,NYP
      DO 230 I3=1,NTP
      SRF(I3,I2)=MAX(ZBOT,SRF(I3,I2))
C
C - REPLACE ALL DATA BY THEIR DECADIC LOGARITHM FOR LOGARITHMIC INCREMENTS
C   BETWEEN CONTOURS
      SRF(I3,I2)=ALOG10(SRF(I3,I2))
 230  CONTINUE
C
C - COMPUTE AND DRAW A SOLID CONTOUR LINE EVERY INTEGER STARTING AT ZERO
      ZPLANE=0.0
      CALL ZBASE(ZPLANE)
      CALL CONLIN(0,'SOLID',LABL,2,8)
      CALL CONMAK(SRF,NTP,NYP,1.0)
      CALL CONTUR(1,LABL,'DRAW')
C
C - COMPUTE AND DRAW A DASHED CONTOUR LINE EVERY INTEGER STARTING AT EVERY
C   LOGARITHM OF THE ELEMENTS OF THE VECTOR LEVEL
      CALL CONLIN(0,'DASH','NOLABELS',1,3)
      DO 240 LV=1,ILN
      ZPLANE=ALOG10(FLOAT(LEVEL(LV)))
      CALL ZBASE(ZPLANE)
      CALL CONMAK(SRF,NTP,NYP,1.0)
      CALL CONTUR(1,'NOLABELS','DRAW')
 240  CONTINUE
C
C - SPECIAL AXIS AND LABEL FOR FFT COORDINATE
      IF ((KSRF.EQ.2.OR.KSRF.EQ.4.OR.KSRF.EQ.6.OR.KSRF.EQ.8)
     1    .AND.NY.GT.8) CALL XISFFT('Y',TORIG,TMAX)
      CALL ENDPL(0)
      RETURN
      END
c
c
c
c
      SUBROUTINE MYCON(DUMMY,IDUMMY)
C
C  This subroutine makes a customized dotted contourline as described
C  in the DISSPLA manual.
c
      DIMENSION RATRAY(2)
      TLENG=0.14
      NMRKSP=2
      RATRAY(1)=1.0/6.0
      RATRAY(2)=5.0/6.0
      CALL MRSCOD(TLENG,NMRKSP,RATRAY)
      RETURN
      END
c
c
c
c
```

```
      SUBROUTINE XISFFT(OORD,YMNDMY,YMXDMY)
c
C  This subroutine calculates the values for the call x/y-graxs which
C  labels the axis of the FFT-variable.
c
C       grfsz = physical size of graphics plots
C       tmax = value at end of time axis
C       torig = value at beginning of time axis
C       tstp = time axis interval
C       wfmax = nice spatial FFT axis end value
C       wforig = nice spatial FFT axis beginning value
C       wfstp = nice spatial FFT axis interval
C       yfmax = value at end of spatial FFT axis
C       yforig = value at beginning of spatial FFT axis
C       yfstp = spatial FFT axis interval
C       ymax = value at end of transverse spatial axis
C       yorig = value at beginning of transverse spatial axis
C       ystp = transverse spatial axis interval
C       uaxor = x- or y-distance of secondary axis from physical origin
C       udiff = difference of original axis end values
C       ulnth = length of customized axis in inches
C       vaxor = x- or y-distance of secondary axis form physical origin
C       xdum = dummy variable holding the x-coordinate of two points
C       ydum = dummy variable holding the y-coordinate of two points
c
      PARAMETER (NT=256,NTP=NT+1,NX=8,NY=128,NYP=NY+1)
C
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION ISRF(8),ITYPE(8),LEVEL(8),CSEC(19,NX),RTYPE(8),
     1          SRF(NTP,NYP),SRFI(NTP,NYP),XDUM(2),YDUM(2)
      COMMON /GRAPHS/ ILN,ISHM,ISRF,ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
     1      GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
     2      YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
     3      ZVAL
C
C - NO FFT-AXIS IN ONE-DIMENSIONAL TRANSIENT CASE
      IF (NY.LE.8) THEN
         WRITE (59,*) 'NO FFT-AXIS WHEN NY.LE.8'
         RETURN
      ENDIF
C
C - WARNING NOT ONE INTERVAL FITS BETWEEN EXTREMA
      IF (WFORIG+WFSTP.GE.WFMAX) WRITE (59,*) 'AXIS ON FFT PLOTS WRONG'
C
C - COMPUTE AXIS LENGTH AND ORIGIN
      UDIFF=YFMAX-YFORIG
      ULNTH=GRFSZ*(WFMAX-WFORIG)/UDIFF
      UAXOR=GRFSZ*(WFORIG-YFORIG)/UDIFF
      VAXOR=0.0
      ORD='Y'
      IF (OORD.EQ.ORD) THEN
C
C - Y-AXIS (IN CONTOUR PLOTS)
C
C - YAXIS TICKMARKS ON THE RIGHT
         NTIK=NINT((WFMAX-WFORIG)/WFSTP)
         XDUM(1)=TMAX-(TMAX-TORIG)/50.0
         XDUM(2)=TMAX
         YDM=WFORIG-WFSTP
         DO 250 ITK=1,NTIK+1
         YDM=YDM+WFSTP
         YDUM(1)=YDM
         YDUM(2)=YDM
         CALL CURVE(XDUM,YDUM,2,0)
 250     CONTINUE
         CALL RESET('YNONUM')
         CALL RESET('YTICKS')
         CALL YGRAXS(WFORIG,WFSTP,WFMAX,ULNTH,
     1        'INVERSE WAVE LENGTH (1/CM)$',100,VAXOR,UAXOR)
      ELSE
         WRITE (59,*)'A FFT'
C
C - X-AXIS (IN CROSS SECTIONAL PLOTS)
         CALL RESET('XNONUM')
         CALL RESET('XTICKS')
C
C - DRAW LINE FOR AXIS
         XDUM(1)=YFORIG
         XDUM(2)=YFMAX
         YDUM(1)=YMNDMY
         YDUM(2)=YMNDMY
         CALL CURVE(XDUM,YDUM,2,0)
         WRITE (59,*)'B FFT'
C
C - YAXIS TICKMARKS ON THE RIGHT
         NTIK=NINT((WFMAX-WFORIG)/WFSTP)
         WRITE (59,*)'C FFT'
         YDUM(1)=YMXDMY-(YMXDMY-YMNDMY)/50.0
         YDUM(2)=YMXDMY
         XDM=WFORIG-WFSTP
         DO 255 ITK=1,NTIK+1
         XDM=XDM+WFSTP
         XDUM(1)=XDM
         XDUM(2)=XDM
         CALL CURVE(XDUM,YDUM,2,0)
 255     CONTINUE
         WRITE (59,*)'D FFT'
C
C - DRAW CUSTOMIZED TICK MARKS
         CALL XGRAXS(WFORIG,WFSTP,WFMAX,ULNTH,
     1        'INVERSE WAVE LENGTH (1/CM)$',100,UAXOR,VAXOR)
      ENDIF
      RETURN
      END
c
c
c
c
      SUBROUTINE CRSSCT(MSRF)
c
C  This subroutine was written by Godehard Hilfer (3/87). It generates
C  cross sectional plots of the data in the two dimensional array(s)
C  srf (srfi).
c
C  Three types of cross sectional plots are available: intensity plots
C  (following statement label 300), phase plots, and amplitude plots
C  (both following statement label 400). When intensity cross sections
C  are called for, this subroutine executes do-loop 390 that does all
C  cross sections specified in row msrf of array cseC and thereafter
C  returns control to the main program. When phase or amplitue cross
C  sections are called for, this subroutine executes do-loop 490 which
C  generates all phase sections specified in row msrf of array csec.
C  Immediately afterwards do-loop 590 is executed which generates all
C  amplitude cross sections that are specified in row msrf+1 of array
C  csec. After this control is returned to the main program.
c
```

```
C  Each type of cross sections is prepared in a similar fashion. In
C  the case of one dimensional data (ny or nt less than 9) only one
C  argument of the arrays srf and srfi is an independent variable
C  the other argument serves as a label to allow distinction between
C  up to eight one dimensional data sets. Which one of these eight
C  data sets is to be graphed is determined by the value of the real
C  part of the element of cseC under consideration (the imaginary
C  part is meaningless in these cases). When nt and ny are larger than
C  8 srf and srfi contain one two-dimensional data set, a surface.
C  Which of the two free variables is to be held constant for
C  cross sectional plots is determined by the imaginary part of the
C  element of cseC under consideration. Therefore, in 2-d cases
C  the imaginary part of the current element of cseC is tested. If it
C  is 2.0 a horizontal cross section (second variable of array(s) srf
C  (srfi) fixed) follows, if it is 1.0 a vertical cross section (first
C  variable of array(s) srf (srfi) fixed) follows, otherwise the next
C  element of cseC will be considered in the same way. For the present
C  graph the headline and axis labels are written onto a new graphics
C  frame, the curve data are computed, the coordinate system is drawn
C  and finally the cross sectional curve itself. If the plot displays
C  FFT data the drawing of the FFT-axis that would be drawn by the call
C  graf will be suppressed in order to avoid the tick mark labels at the
C  very end of this axis which would exhibit messy numbers. In the
C  place of the suppressed axis a 'secondary' (DISSPLA nomenclature)
C  axis will be drawn immediately after the cross sectional curve
C  is drawn. This secondary axis exhibits 'nicely' valued tick marks.
c
C  The cross sectional curves represent the functional values at the
C  grid point iseC that is clossest to the locations specified by the
C  real part of the current element of csec. While the data of the
C  intensity and amplitude plots are readily available from the
C  array(s) srf (srfi) the data for the phase sections have to be
C  calculated first by this subroutine.
c
C  The phase data are calculated as follows. The field magnitude at the
C  fixed grid point iseC is computed. If its maximum is less than
C  10**(-30) the field information is determined unreliable and no
C  phase curve will be drawn. Furthermore all locations where the
C  magnitude is less than the maximum magnitude divided by 10**8 are
C  determined as points of unreliable field information and will
C  exhibit no phase curve point. The arctangent of the ratio of the
C  imaginary to real field amplitudes provides the raw phase data. It
C  is assumed that the numerical resolution of RAM2D1 is sufficient to
C  provide raw phase data that do not vary by more than +/- pi from
C  grid point to grid point. The first raw data point falls within
C  +/- pi of zero phase. All consecutive raw data points are tested if
C  they were reached by a phase change that implies a crossing of the
C  negative real axis of the amplitude vector in which case 2 pi will
C  be added or subtracted to all following phase points depending on an
C  implied phase windup or wind-down. By this method phase variations
C  over multiples of 2 pi can be followed. In case of intermittened
C  unreliable data points the next reliable phase is placed within the
C  same 2 pi interval as the previous reliable phase.
c
C                                --variables--
c
C      cseC = 2-dim array with cross sectional information
C      grfsz = physical size of graphics plots
C      iseC = srf(i) grid point corresponds clossest to real part of
C             csec
C      k = index in do-loops 390,490,590
C      k1 = index in do-loops 320,375,410,460,510,520,555,560
C      k2 = index in do-loops 330,380,411,465,530,565
```

```
C       k3 = index in do-loops 423,473
C       k4 = index in do-loops 429,475
C       kcnt = index when calculating and plotting phase data
C       lpi = multiples of 2 pi counter
C       msrf = index number of surface of which cross sections are drawn
C       nab = index of the first of a string of reliable phase data
C             points
C       nan = index of the last of a string of relaible phase data points
C       necveC = data switch for subroutine nyaxis
C       npoints = number of data points to be drawn
C       nseC = number of elements tested in rows of csec
C       nsrf = index number of amplitude surface of which cross sections
C              are drawn
C       nt = see RAM2D1
C       ntp = nt+1
C       ny = see RAM2D1
C       nyhp = ny/2+1
C       nyp = ny+1
C       phasdf = test variable deciding phase axis interval
C       phasmx = phase axis end value
C       phasor = phase axis beginning value
C       phastp = phase axis interval
C       phsmxi = integer clossest to phasmx
C       phsori = integer clossest to phsori
C       pi = 3.14159265358979
C       psik = phase being tested for 2 pi interval
C       psip = previous phase referencing in 2 pi interval test
C       rdt = step size in time
C       rdy = step size in transverse spatial variable y
C       rdyx = grid point spacing on horizontal axis
C       scmem = array containing initial longitudinal invariant data
C       sctold = last reliable phase before unreliable phase data
C       seci = imaginary part of current cseC element
C       secr = real part of current cseC element
C       secti = vector containing phase data or imaginary amplitude data
C       sectn = vector containing intensity data, magnitude data, or real
C               amplitude data
C       srf = source data array from main program
C       srfi = source data array from main program (imaginary part)
C       tm1 = time coordinate lower limit
C       tm2 = time coordinate upper limit
C       tmax = value at end of time axis
C       torig = value at beginning of time axis
C       tstp = time axis interval
C       wfmax = nice spatial FFT axis end value
C       wforig = nice spatial FFT axis beginning value
C       wfstp = nice spatial FFT axis interval
C       wmax = nice vertical axis end value
C       worig = nice vertical axis beginning value
C       wstp = nice vertical axis interval
C       yfmax = value at end of spatial FFT axis
C       yforig = value at beginning of spatial FFT axis
C       yfstp = spatial FFT axis interval
C       ymax = value at end of transverse spatial axis
C       yorig = value at beginning of transverse spatial axis
C       ystp = transverse spatial axis interval
C       xi = imaginary amplitude value
C       xmx = section maximum
C       xr = real amplitude value
C       xthrsh = fraction of intensity below which data are considered
C                unreliable
C       xx = vector containing physical x-axis values for plotting
C       xdum = dummy variable holding the x-coordinate of two points
```

```
C      ydum = dummy variable holding the y-coordinate of two points
C      ym1 = y-coordinate lower limit
C      ym2 = y-coordinate upper limit
C      ym2m1 = ym2-ym1
C      zval = current z-location
C ---------------------------------------------------------------------
c
      PARAMETER (NP=10,NT=256,NTP=NT+1,NX=8,NY=128,NYH=NY/2,NYHP=NYH+1,
     1          NYP=NY+1,NTPY=NT+NY)
C
      IMPLICIT COMPLEX(A-E,Q)
      DIMENSION ISRF(8),ITYPE(8),IWHEN(NYHP),LEVEL(8),CSEC(19,NX),
     1          PSMEM(NTPY,8),PPMEM(NTPY,8),RTYPE(8),SCMEM(NTPY,8),
     2          SECTI(NTPY),SECTJ(NTPY),SECTN(NTPY),SRF(NTP,NYP),
     3          SRFI(NTP,NYP),TIK(NY),XDUM(2),XX(NTPY),YDUM(2)
      COMMON /GRAPHS/ ILN,ISHM,ISRF ITYPE,LEVEL,NDEC,NHYP,NSEC,CSEC,
     1       GRFSZ,PI,RTYPE,SRF,SRFI,TMAX,TORIG,TSTP,YFMAX,YFORIG,
     2       YFSTP,YMAX,YORIG,YSTP,WFMAX,WFORIG,WFSTP,ZBOT,ZMAX,ZSTEP,
     3       ZVAL
      COMMON /NUM/ RDT,RDY,RDYF,TM1,TM2,YM1,YM2,YM2M1
      COMMON WORK(25000)
C
C - ERROR CONDITIONS
      IF (MSRF.LT.1.OR.MSRF.GT.19) THEN
         WRITE (59,*) 'MSRF = ',MSRF,' IN CRSSCT OUT OF RANGE'
         RETURN
      ENDIF
C
      IF (NY.LE.8) THEN
         GO TO (290,290,290, 280,280,280, 290,290,290,
     1         280,280,280, 290,290,290, 280,280,280, 290) MSRF
  280    RETURN
  290    CONTINUE
      ENDIF
      GO TO (300,400,400, 300,400,400, 300,400,400,
     1       300,400,400, 300,400,400, 300,400,400, 300) MSRF
  300 CONTINUE
C
C ---- CROSS SECTIONS OF INTENSITY SURFACES OR MATERIAL EXCITATION
C
C --- CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
      DO 390 K=1,NSEC
      SECR=REAL(CSEC(MSRF,K))
C
C - ONE-DIMENSIONAL CASES
      IF (NY.LE.8) THEN
         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 390
         GO TO 310
      ELSE IF (NT.LE.8) THEN
         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 390
         GO TO 340
      ENDIF
C
C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC-
C   ELEMENT IS EQUAL TO 1.0 OR 2.0;
C   OTHERWISE GO TO NEXT LOOP COUNTER K, I.E. NEXT ELEMENT OF LINE MSRF
C   IN ARRAY CSEC
      SECI=AIMAG(CSEC(MSRF,K))
      IF (SECI.GT.0.9.AND.SECI.LT.1.1) GO TO 340
      IF (SECI.GT.1.9.AND.SECI.LT.2.1) GO TO 310
      GO TO 390
  310 CONTINUE
C
```

```
C -- HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); INTENSITY
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL AREA2D(GRFSZ,GRFSZ)
      CALL INTAXS
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO (311,390,390, 312,390,390, 313,390,390,
     1      314,390,390, 315,390,390, 316,390,390, 317) MSRF
 311  CALL HEADIN('RAMAN PUMP: INTENSITY$',100,1.5,1)
      GO TO 321
 312  CALL HEADIN('RAMAN PUMP: MODE INTENSITY$',100,1.5,1)
      GO TO 321
 313  CALL HEADIN('RAMAN STOKES: INTENSITY$',100,1.5,1)
      GO TO 321
 314  CALL HEADIN('RAMAN STOKES: MODE INTENSITY$',100,1.5,1)
      GO TO 321
 315  CALL HEADIN('RAMAN MAT. EXC.: INTENSITY$',100,1.5,1)
      GO TO 321
 316  CALL HEADIN('RAMAN MAT. EXC.: MODE INTENSITY$',100,1.5,1)
      GO TO 321
 317  CALL HEADIN('RAMAN AMPLIFIER Z-INVARIANT$',100,1.5,1)
 321  CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      CALL XNAME('TIME (PICO-SECONDS)$',100)
      IF (MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16) THEN
         CALL YNAME('INTENSITY IN MODE KY$',100)
         CALL MESSAG('KY =  $',100,4.2,7.1)
         IPLACE=2
         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
         CALL REALNO(SECR,IPLACE,4.7,7.1)
         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
         ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
      ELSE
         IF (MSRF.EQ.19.) THEN
            CALL YNAME('LONGITUDINAL INVARIANT$',100)
            IF (ZVAL.GT.ZSTEP) THEN
               CALL MESSAG('DASHED = INVARIANT AT Z=0$',100,0.1,7.35)
            ENDIF
         ELSE
            CALL YNAME('INTENSITY$',100)
         ENDIF
         IF (NY.LE.8) THEN
            ISEC=NINT(SECR)
            GO TO (322,323,324,325) ITYPE(ISEC)
 322        CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
            GO TO 326
 323        CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
            GO TO 326
 324        CALL MESSAG('LORENTZIAN  , EXP = $',100,0.1,7.1)
            GO TO 326
 325        CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)
 326        CONTINUE
            IF (ITYPE(ISEC).NE.2) THEN
               XRTYPE=RTYPE(ISEC)
               IPLACE=2
               IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)
     1         IPLACE=-2
               CALL REALNO(XRTYPE,IPLACE,2.4,7.1)
            ENDIF
            IF (NY.GT.1) THEN
              CALL MESSAG('CASE$',100,4.0,7.1)
              CALL INTNO(ISEC,4.7,7.1)
            ENDIF
            CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
          ELSE
            CALL MESSAG('Y =  $',100,4.2,7.1)
            IPLACE=2
            IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
            CALL REALNO(SECR,IPLACE,4.7,7.1)
            CALL MESSAG('2 - DIM.$',100,5.9,7.35)
            ISEC=INT((SECR+RDY*NYHP)/RDY)
          ENDIF
      ENDIF
C
C - CROSS SECTION DATA
      DO 327 K1=1,NT
      SECTN(K1)=SRF(K1,ISEC)
      XX(K1)=TM1+RDT*(K1-1)
 327  CONTINUE
C
C - TOTAL INTENSITY INTEGRAL IN 1-D
      IF (NY.LE.8) THEN
         TOTI=0.0
         DO 329 K1=1,NT
         TOTI=TOTI+SECTN(K1)
 329     CONTINUE
         TOTI=TOTI*RDT
         IF (ZVAL.LT.ZSTEP) THEN
C
C - INTEGRAL VALUE ONTO GRAPH WHEN Z=0
            IF (MSRF.EQ.1.AND.TOTI.GT.0.0) THEN
               TTPSTO=TOTI
               CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
               IPLACE=4
               IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
               CALL REALNO(TOTI,IPLACE,1.4,6.7)
            ENDIF
            IF (MSRF.EQ.7.AND.TOTI.GT.0.0) THEN
               TTSSTO=TOTI
               CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
               IPLACE=4
               IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
               CALL REALNO(TOTI,IPLACE,1.4,6.7)
            ENDIF
            IF (MSRF.EQ.7) TTSSTO=TOTI
         ELSE
C
C - DEPLETION/GAIN VALUE ONTO GRAPH WHEN Z>0
            IF (MSRF.EQ.1.AND.TTPSTO.GT.0.0) THEN
               RINTEG=TOTI/TTPSTO
               CALL MESSAG('DEPLETION= $',100,0.1,6.7)
               IPLACE=4
               IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
               CALL REALNO(RINTEG,IPLACE,1.4,6.7)
            ENDIF
            IF (MSRF.EQ.7.AND.TTSSTO.GT.0.0) THEN
               RINTEG=TOTI/TTSSTO
               CALL MESSAG('GAIN= $',100,0.1,6.7)
               IPLACE=4
               IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
               CALL REALNO(RINTEG,IPLACE,0.7,6.7)
            ENDIF
         ENDIF
      ENDIF
C
C - MEMORIZE INVARIANT AT Z=0
      IF (MSRF.EQ.19.AND.ZVAL.LT.ZSTEP) THEN
         DO 335 K1=1,NT
         SCMEM(K1,K)=SECTN(K1)
 335     CONTINUE
      ENDIF
C
C - DRAW COORDINATE SYSTEM
      NECLEC=1
      WSTP=-1.0
      WORIG=0.0
      CALL NYSXIS(SECTN,NT,NECLEC,WORIG,WSTP,WMAX)
      CALL GRAF(TORIG,TSTP,TMAX,WORIG,WSTP,WMAX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
      XDUM(1)=TORIG
      XDUM(2)=TMAX
      YDUM(1)=WMAX
      YDUM(2)=WMAX
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((TMAX-TORIG)/TSTP)
      YDUM(1)=WMAX
      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
      XDM=TORIG
      DO 336 ITK=1,NTIK-1
      XDM=XDM+TSTP
      XDUM(1)=XDM
      XDUM(2)=XDM
      CALL CURVE(XDUM,YDUM,2,0)
 336  CONTINUE
      XDUM(1)=TMAX
      XDUM(2)=TMAX
      YDUM(1)=WMAX
      YDUM(2)=WORIG
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((WMAX-WORIG)/WSTP)
      XDUM(1)=TMAX-(TMAX-TORIG)/50.0
      XDUM(2)=TMAX
      YDM=WORIG
      DO 337 ITK=1,NTIK-1
      YDM=YDM+WSTP
      YDUM(1)=YDM
      YDUM(2)=YDM
      CALL CURVE(XDUM,YDUM,2,0)
 337  CONTINUE
C
C - DRAW CROSS SECTION CURVE
      NPOINTS=NT
      CALL CURVE(XX,SECTN,NPOINTS,0)
C
C - DRAW Z=0 INVARIANT FOR COMPARISON
      IF (MSRF.EQ.19.AND.ZVAL.GT.ZSTEP) THEN
         DO 338 K1=1,NT
         SECTN(K1)=SCMEM(K1,K)
 338     CONTINUE
         CALL DASH
         CALL CURVE(XX,SECTN,NPOINTS,0)
         CALL RESET('DASH')
      ENDIF
```

```
C
C - END OF PLOT
      CALL ENDPL(0)
      GO TO 390
  340 CONTINUE
C
C -- VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); INTENSITY
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL AREA2D(GRFSZ,GRFSZ)
      CALL INTAXS
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO (355,390,390, 356,390,390, 357,390,390,
     1       358,390,390, 359,390,390, 360,390,390, 361) MSRF
  355 CALL HEADIN('RAMAN PUMP: INTENSITY$',100,1.5,1)
      GO TO 371
  356 CALL HEADIN('RAMAN PUMP: INTENSITY (FFT)$',100,1.5,1)
      GO TO 371
  357 CALL HEADIN('RAMAN STOKES: INTENSITY$',100,1.5,1)
      GO TO 371
  358 CALL HEADIN('RAMAN STOKES: INTENSITY (FFT)$',100,1.5,1)
      GO TO 371
  359 CALL HEADIN('RAMAN MAT. EXC.: INTENSITY$',100,1.5,1)
      GO TO 371
  360 CALL HEADIN('RAMAN MAT. EXC.: INTENSITY (FFT)$',100,1.5,1)
      GO TO 371
  361 CALL HEADIN('RAMAN LONGITUDINAL INVARIANT$',100,1.5,1)
  371 CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      IF (MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16) THEN
         CALL XNONUM
         CALL XTICKS(0)
         CALL YNAME('FFT INTENSITY$',100)
         RDYX=RDYF
         XORIG=YFORIG
         XSTP=YFSTP
         XMAX=YFMAX
      ELSE
         CALL XNAME('Y-DIMENSION (CM)$',100)
         IF (MSRF.EQ.19.) THEN
            CALL YNAME('LONGITUDINAL INVARIANT$',100)
            IF (ZVAL.GT.ZSTEP) THEN
               CALL MESSAG('DASHED = INVARIANT AT Z=0$',100,0.1,7.35)
            ENDIF
         ELSE
            CALL YNAME('INTENSITY$',100)
         ENDIF
         RDYX=RDY
         XORIG=YORIG
         XSTP=YSTP
         XMAX=YMAX
      ENDIF
      IF (NT.LE.8) THEN
         ISEC=NINT(SECR)
         CALL MESSAG('EXPON., NHYP = $',100,0.1,7.1)
         CALL INTNO(NHYP,2.0,7.1)
         CALL MESSAG('1 - D STA.$',100,5.9,7.35)
         IF (NT.GT.1) THEN
```

```
                CALL MESSAG('CASE$',100,4.0,7.1)
                CALL INTNO(ISEC,4.7,7.1)
             ENDIF
          ELSE
             ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
             CALL MESSAG('T =  $',100,4.2,7.1)
             IPLACE=2
             IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
             CALL REALNO(SECR,IPLACE,4.7,7.1)
             CALL MESSAG('2 - DIM.$',100,5.9,7.35)
          ENDIF
C
C - CROSS SECTION DATA
          DO 373 K1=1,NY
          SECTN(K1)=SRF(ISEC,K1)
          XX(K1)=XORIG+RDYX*(K1-1)
  373     CONTINUE
C
C - TOTAL INTENSITY INTEGRAL IN 1-D
          IF (NT.LE.8) THEN
             TOTI=0.0
             DO 374 K1=1,NY
             TOTI=TOTI+SECTN(K1)
  374        CONTINUE
             TOTI=TOTI*RDY
             IF (ZVAL.LT.ZSTEP) THEN
C
C - INTEGRAL VALUE ONTO GRAPH WHEN Z=0
                IF (MSRF.EQ.1.AND.TOTI.GT.0.0) THEN
                   TTPSTO=TOTI
                   CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
                   IPLACE=4
                   IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
                   CALL REALNO(TOTI,IPLACE,1.4,6.7)
                ENDIF
                IF (MSRF.EQ.7.AND.TOTI.GT.0.0) THEN
                   TTSSTO=TOTI
                   CALL MESSAG('INTEGRAL= $',100,0.1,6.7)
                   IPLACE=4
                   IF (TOTI.GT.9999.0.OR.TOTI.LT.0.01) IPLACE=-2
                   CALL REALNO(TOTI,IPLACE,1.4,6.7)
                ENDIF
                IF (MSRF.EQ.7) TTSSTO=TOTI
             ELSE
C
C - DEPLETION/GAIN VALUE ONTO GRAPH WHEN Z>0
                IF (MSRF.EQ.1.AND.TTPSTO.GT.0.0) THEN
                   RINTEG=TOTI/TTPSTO
                   CALL MESSAG('DEPLETION= $',100,0.1,6.7)
                   IPLACE=4
                   IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
                   CALL REALNO(RINTEG,IPLACE,1.4,6.7)
                ENDIF
                IF (MSRF.EQ.7.AND.TTSSTO.GT.0.0) THEN
                   RINTEG=TOTI/TTSSTO
                   CALL MESSAG('GAIN= $',100,0.1,6.7)
                   IPLACE=4
                   IF (RINTEG.GT.9999.0.OR.RINTEG.LT.0.01) IPLACE=-2
                   CALL REALNO(RINTEG,IPLACE,0.7,6.7)
                ENDIF
             ENDIF
          ENDIF
C
```

```
C - MEMORIZE INVARIANT AT Z=0
      IF (MSRF.EQ.19.AND.ZVAL.LT.ZSTEP) THEN
         DO 375 K1=1,NY
         SCMEM(K1,K)=SECTN(K1)
 375     CONTINUE
      ENDIF
      IF (MSRF.EQ.10) THEN
C
C - COMPUTE TOTAL INTENSITY OF STOKES BEAM IN K-SPACE
         TIK(1)=0.0
         DO 376 J=2,NY
         TIK(J)=TIK(J-1)+SECTN(J)*RDYF
 376     CONTINUE
         TIKBMX=TIK(NY)
         WRITE (59,*)'TIKBMX= ',TIKBMX
         CALL MESSAG('TOT. INT. = $',100,0.2,6.7)
         IPLACE=2
         IF (ABS(TIKBMX).GT.9999.0.OR.ABS(TIKBMX).LT.0.01) IPLACE=-2
         CALL REALNO(TIKBMX,IPLACE,1.6,6.7)
         WRITE (59,*)'TOTAL INTENSITY OF STOKES IN K= ',TIKBMX
C
C - COMPUTE K-WIDTH OF STOKES (LINEAR INTERPOLATION)
         DO 377 J=1,NY
         TIK(J)=ABS((2.0*TIK(J)-TIKBMX)/TIKBMX)-2.0/PI
 377     CONTINUE
         CALL WHENFLE(NY,TIK(1),1,0.0,IWHEN,NVAL)
         IWN1=IWHEN(1)+JL-1
         IWN2=IWHEN(NVAL)+JL-1
         IF (IWN2-IWN1.LT.4) THEN
            WRITE (59,*) 'INSUFFICIENT RESOLUTION FFT BEAM ',IB
         ENDIF
         WRITE (59,*)'IWHEN = ',IWHEN
         YKL=YFORIG+RDYF*(IWN1+TIK(IWN1)/(TIK(IWN1-1)-TIK(IWN1)))
         YKR=YFORIG+RDYF*(IWN2-TIK(IWN2)/(TIK(IWN2+1)-TIK(IWN2)))
         WDTHKB=YKR-YKL
         CALL MESSAG(' K WIDTH  = $',100,4.2,6.7)
         IPLACE=2
         IF (ABS(WDTHKB).GT.9999.0.OR.ABS(WDTHKB).LT.0.01) IPLACE=-2
         CALL REALNO(WDTHKB,IPLACE,5.6,6.7)
      ENDIF
C
C - DRAW COORDINATE SYSTEM
      NECLEC=1
      WSTP=-1.0
      WORIG=0.0
      CALL NYSXIS(SECTN,NY,NECLEC,WORIG,WSTP,WMAX)
      CALL GRAF(XORIG,XSTP,XMAX,WORIG,WSTP,WMAX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
      XDUM(1)=XORIG
      XDUM(2)=XMAX
      YDUM(1)=WMAX
      YDUM(2)=WMAX
      CALL CURVE(XDUM,YDUM,2,0)
      IF (NY.GT.8.AND.(MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16)) GOTO 380
      NTIK=NINT((XMAX-XORIG)/XSTP)
      YDUM(1)=WMAX
      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
      XDM=XORIG
      DO 378 ITK=1,NTIK-1
      XDM=XDM+XSTP
      XDUM(1)=XDM
      XDUM(2)=XDM
      CALL CURVE(XDUM,YDUM,2,0)
 378  CONTINUE
 380  CONTINUE
      XDUM(1)=XMAX
      XDUM(2)=XMAX
      YDUM(1)=WMAX
      YDUM(2)=WORIG
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((WMAX-WORIG)/WSTP)
      XDUM(1)=XMAX-(XMAX-XORIG)/50.0
      XDUM(2)=XMAX
      YDM=WORIG
      DO 382 ITK=1,NTIK-1
      YDM=YDM+WSTP
      YDUM(1)=YDM
      YDUM(2)=YDM
      CALL CURVE(XDUM,YDUM,2,0)
 382  CONTINUE
C
C - DRAW CROSS SECTION CURVE
      NPOINTS=NY
      CALL CURVE(XX,SECTN,NPOINTS,0)
C
C - DRAW Z=0 INVARIANT FOR COMPARISON
      IF (MSRF.EQ.19.AND.ZVAL.GT.ZSTEP) THEN
         DO 385 K1=1,NY
         SECTN(K1)=SCMEM(K1,K)
 385     CONTINUE
         CALL DASH
         CALL CURVE(XX,SECTN,NPOINTS,0)
         CALL RESET('DASH')
      ENDIF
C
C - SPECIAL AXIS AND LABEL FOR FFT COORDINATE
      IF (NY.GT.8.AND.(MSRF.EQ.4.OR.MSRF.EQ.10.OR.MSRF.EQ.16))
     1   CALL XISFFT('X',WORIG,WMAX)
      WRITE (59,*)'END OF STATIONARY INTENSITY PLOT'
C
C - END OF PLOT
      CALL ENDPL(0)
 390  CONTINUE
       RETURN
C
C ---- END OF INTENSITY SECTION
C
 400  CONTINUE
C
C ---- PHASE AND AMPLITUDE SECTIONS
C
C --- CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
      DO 490 K=1,NSEC
      SECR=REAL(CSEC(MSRF,K))
C
C - ONE DIMENSIONAL CASES
      IF (NY.LE.8) THEN
         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 490
         GO TO 401
      ELSE IF (NT.LE.8) THEN
         IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 490
         GO TO 450
      ENDIF
C
C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC-
```

```
C   ELEMENT IS EQUAL TO 1.0 OR 2.0;
C   OTHERWISE GO TO NEXT LOOP COUNTER K, I.E. NEXT ELEMENT OF LINE MSRF
C   IN ARRAY CSEC
      SECI=AIMAG(CSEC(MSRF,K))
      IF (SECI.GT.0.9.AND.SECI.LT.1.1) GO TO 450
      IF (SECI.GT.1.9.AND.SECI.LT.2.1) GO TO 401
      GO TO 490
 401  CONTINUE
C
C -- HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); PHASE
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL INTAXS
      CALL MX1ALF('STANDARD','!')
      CALL MX2ALF('L/CGRK','#')
      CALL AREA2D(GRFSZ,GRFSZ)
      IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
         CALL MESSAG('SOLID = INTERFER.$',100,0.1,7.35)
         CALL MESSAG('DASHED = ACTUAL$',100,2.3,7.35)
      ENDIF
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO (490,402,490, 490,403,490, 490,404,490,
     1       490,405,490, 490,406,490, 490,407,490) MSRF
 402  CALL HEADIN('RAMAN PUMP: PHASE$',100,1.5,1)
      GO TO 409
 403  CALL HEADIN('RAMAN PUMP: MODE PHASE$',100,1.5,1)
      GO TO 409
 404  CALL HEADIN('RAMAN STOKES: PHASE$',100,1.5,1)
      GO TO 409
 405  CALL HEADIN('RAMAN STOKES: MODE PHASE$',100,1.5,1)
      GO TO 409
 406  CALL HEADIN('RAMAN MAT. EXC.: PHASE$',100,1.5,1)
      GO TO 409
 407  CALL HEADIN('RAMAN MAT. EXC.: MODE PHASE$',100,1.5,1)
 409  CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      CALL XNAME('TIME (PICO-SECONDS)$',100)
      IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) THEN
         CALL YNAME('MODE PHASE (MULTIPLES OF #PI)$',100)
         CALL MESSAG('KY =  $',100,4.2,7.1)
         IPLACE=2
         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
         CALL REALNO(SECR,IPLACE,4.7,7.1)
         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
         ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
      ELSE
         CALL YNAME('PHASE (MULTIPLES OF #PI)$',100)
         IF (NY.LE.8) THEN
            CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
            ISEC=NINT(SECR)
            GO TO (412,413,414,415) ITYPE(ISEC)
 412        CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
            GO TO 416
 413        CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
            GO TO 416
 414        CALL MESSAG('LORENTZIAN  , EXP = $',100,0.1,7.1)
            GO TO 416
 415        CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)
 416        CONTINUE
            IF (ITYPE(ISEC).NE.2) THEN
               XRTYPE=RTYPE(ISEC)
               IPLACE=2
               IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)
     1         IPLACE=-2
               CALL REALNO(XRTYPE,IPLACE,2.4,7.1)
            ENDIF
            IF (NY.GT.1) THEN
               CALL MESSAG('CASE$',100,4.0,7.1)
               CALL INTNO(ISEC,4.7,7.1)
            ENDIF
         ELSE
            CALL MESSAG('Y =  $',100,4.2,7.1)
            IPLACE=2
            IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
            CALL REALNO(SECR,IPLACE,4.7,7.1)
            CALL MESSAG('2 - DIM.$',100,5.9,7.35)
            ISEC=INT((SECR+RDY*NYHP)/RDY)
         ENDIF
      ENDIF
C
C - MAGNITUDE DATA, ABSCISSA VECTOR
      DO 418 K1=1,NT
      XR=SRF(K1,ISEC)
      XI=SRFI(K1,ISEC)
      SECTN(K1)=SQRT(XR*XR+XI*XI)
      XX(K1)=TM1+RDT*(K1-1)
 418  CONTINUE
C
C - UNCERTAIN PHASE THRESHOLD
      XMX=SECTN(ISMAX(NT,SECTN,1))
      IF (XMX.LT.1.0E-30) THEN
         WRITE (59,*) 'note: UNRELIABLE PHASE, MAGNITUDES ARE ZERO'
         GO TO 490
      ENDIF
      XTHRSH=XMX/1.0E8
C
C -- CALCULATE PHASE DATA
C
C - PHASE OF FIRST DATA POINT WITHIN +/- PI OF ZERO PHASE
      SCTOLD=0.0
C
C - INITIALIZE LOOP VARIABLES
      NAB=1
      NAN=0
      KINCR=0
C
C -- LOOP OVER ALL GRID POINTS; KINCR LOOP COUNTER
 420  CONTINUE
      KINCR=KINCR+1
C
C - CLEAR VECTOR FOR PHASE DATA
      SECTI(KINCR)=0.0
C
C - FIND STRING OF GRID POINTS (NAB TO NAN) WHERE MAGNITUDE OF FIELD
C   DATA EXCEEDS THRESHOLD
      IF (SECTN(KINCR).GE.XTHRSH) THEN
         NAN=KINCR
C
C - PLACE MARKER (SECTN=-1.0) WHERE FIELD MAGNITUDE IS BELOW THRESHOLD
C   (UNCERTAIN PHASE INFORMATION)
      ELSE
```

```
            SECTN(KINCR)=-1.0
            IF (NAN.GE.NAB) THEN
C
C - EXIT LOOP TEMPORARILY TO CALCULATE PHASE DATA FOR STRING OF GRID
C   POINTS NAB TO NAN
              GO TO 421
            ELSE
C
C - CURRENT DATA POINT STILL UNCERTAIN; INCREMENT NAB (BEGINNING OF NEXT
C   STRING)
              NAB=KINCR+1
            ENDIF
          ENDIF
C
C -- END LOOP OR CONTINUE IN LOOP UNTIL NT
          IF (KINCR.LT.NT) GO TO 420
C
C - SKIP PHASE CALCULATION, LAST DATA POINTS ARE UNCERTAIN
          IF (NAN.LT.NAB) GO TO 432
C
C - CALCULATE RAW PHASE MODULO 2*PI
  421     CONTINUE
          DO 423 K3=NAB,NAN
          SECTI(K3)=ATAN2(SRFI(K3,ISEC),SRF(K3,ISEC))/PI
  423     CONTINUE
C
C - CALCULATE EXACT PHASE KEEPING TRACK OF MULIPLES OF 2*PI COUNTED BY
C   LPI
          LPI=0
          IF (NAN.EQ.NAB) THEN
C
C - SINGLE DATA POINT
            SECTI(NAN)=SECTI(NAN)+SCTOLD
            GO TO 431
          ENDIF
C
C - PHASE OF FIRST DATA POINT IN STRING
          PSIP=SECTI(NAB)
          SECTI(NAB)=PSIP+SCTOLD
C
C - PHASE OF FOLLOWING DATA POINTS IN STRING
          DO 429 K4=NAB+1,NAN
          PSIK=SECTI(K4)
          IF (PSIP.GE.0.0) THEN
C
C - INCREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
C   THE PREVIOUS POINT PSIK (WHICH WAS POSITIVE)
            IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI+2
          ELSE
C
C - DECREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
C   THE PREVIOUS POINT PSIK (WHICH WAS NEGATIVE)
            IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI-2
          ENDIF
C
C - EXACT PHASE
          SECTI(K4)=PSIK+LPI+SCTOLD
C
C - CURRENT RAW PHASE BECOMES PREVIOUS RAW PHASE NEXT TIME THROUGH THE
C   LOOP
          PSIP=PSIK
  429     CONTINUE
  431     CONTINUE
```

```
C
C - STORE PHASE OF LAST DATA POINT AS REFERENCE VALUE FOR NEXT STRING OF
C   RELIABLE DATA
      SCTOLD=SECTI(NAN)
C
C - INCREMENT LABEL OF BEGINNING OF NEXT STRING
         NAB=KINCR+1
C
C - FIND NEXT STRING OF PHASE DATA
      IF (NAN.LT.NT) GO TO 420
 432  CONTINUE
C
C - PLOT COORDINATE SYSTEM
      IF ((MSRF.EQ.2.OR.MSRF.EQ.8).AND.ZVAL.LT.ZSTEP) THEN
C
C - MEMORIZE ORIGINAL PHASE
         DO 433 I3=1,NT
         IF (MSRF.EQ.2) THEN
            PPMEM(I3,K)=SECTI(I3)
         ELSE
            PSMEM(I3,K)=SECTI(I3)
         ENDIF
 433     CONTINUE
      ELSE
C
C - INTERFEROMETRIC PHASE (CURRENT PHASE MINUS ORIGINAL PHASE)
         DO 434 I3=1,NT
         IF (MSRF.EQ.2) THEN
            SECTJ(I3)=SECTI(I3)-PPMEM(I3,K)
         ELSE
            SECTJ(I3)=SECTI(I3)-PSMEM(I3,K)
         ENDIF
 434     CONTINUE
      ENDIF
C
C - SCALE AXIS BY COMBINATION OF BOTH CURVES
      NECLEC=1
      PHASTP=0.0
      CALL NYSXIS(SECTJ,NT,NECLEC,PHASOR,PHASTP,PHASMX)
      NECLEC=0
      CALL NYSXIS(SECTI,NT,NECLEC,PHASOR,PHASTP,PHASMX)
C
C - MAKE FRACTIONAL Y-AXIS LIMITS INTEGRAL
      IF (PHASTP.LE.1.0) THEN
         PHSORI=ANINT(PHASOR)
         PHSMXI=ANINT(PHASMX)
         IF (PHASOR.LE.PHSORI) THEN
            PHASOR=PHSORI-1.0
         ELSE
            PHASOR=PHSORI
         ENDIF
         IF (PHASMX.GE.PHSMXI) THEN
            PHASMX=PHSMXI+1.0
         ELSE
            PHASMX=PHSMXI
         ENDIF
C
C - SMALLEST Y-INTERVAL SIZES SHOULD BE PI/4 AND PI/2
         PHASDF=PHASMX-PHASOR
         IF (PHASDF.LE.2.0) THEN
            PHASTP=0.25
         ELSE IF (PHASDF.LE.4.0) THEN
            PHASTP=0.5
```

```
          ENDIF
       ENDIF
       CALL GRAF(TORIG,TSTP,TMAX,PHASOR,PHASTP,PHASMX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
       XDUM(1)=TORIG
       XDUM(2)=TMAX
       YDUM(1)=PHASMX
       YDUM(2)=PHASMX
       CALL CURVE(XDUM,YDUM,2,0)
       NTIK=NINT((TMAX-TORIG)/TSTP)
       YDUM(1)=PHASMX
       YDUM(2)=PHASMX-(PHASMX-PHASOR)/50.0
       XDM=TORIG
       DO 435 ITK=1,NTIK-1
       XDM=XDM+TSTP
       XDUM(1)=XDM
       XDUM(2)=XDM
       CALL CURVE(XDUM,YDUM,2,0)
  435  CONTINUE
       XDUM(1)=TMAX
       XDUM(2)=TMAX
       YDUM(1)=PHASMX
       YDUM(2)=PHASOR
       CALL CURVE(XDUM,YDUM,2,0)
       NTIK=NINT((PHASMX-PHASOR)/PHASTP)
       XDUM(1)=TMAX-(TMAX-TORIG)/50.0
       XDUM(2)=TMAX
       YDM=PHASOR
       DO 436 ITK=1,NTIK-1
       YDM=YDM+PHASTP
       YDUM(1)=YDM
       YDUM(2)=YDM
       CALL CURVE(XDUM,YDUM,2,0)
  436  CONTINUE
C
C - PLOT PHASE CURVE SEGMENTS; RESET COUNTERS
       NPOINTS=0
       KINCR=0
C
C -- LOOP OVER DATA POINTS; LOOP COUNTER KINCR
  437  CONTINUE
       KINCR=KINCR+1
       IF (SECTN(KINCR).LT.-0.99) THEN
C
C - UNRELIABLE PHASE MARKER ENCOUNTERED; PLOT DATA STRING OF LENGTH
C   NPOINTS
          IF (NPOINTS.GT.0) GO TO 438
       ELSE
C
C - INCREMENT DATA STRING COUNTER; PUSH RELIABLE DATA TO FRONT OF VECTOR
C   FOR PLOTTING
          NPOINTS=NPOINTS+1
          SECTI(NPOINTS)=SECTI(KINCR)
          IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
             SECTJ(NPOINTS)=SECTJ(KINCR)
          ENDIF
          XX(NPOINTS)=XX(KINCR)
       ENDIF
C
C -- END LOOP OR CONTINUE IN LOOP UNTIL NT
       IF (KINCR.LT.NT) GO TO 437
C
```

```
C - LAST DATA POINTS UNRELIABLE; END PHASE PLOTTING
      IF (NPOINTS.EQ.0) GO TO 440
  438 CONTINUE
C
C - PLOT DATA STRING; RESET DATA COUNTER
C - DRAW INTERFEROMETRIC PHASE SOLID
      IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
         CALL CURVE(XX,SECTJ,NPOINTS,0)
      ENDIF
C
C - DRAW CURRENT PHASE DASHED
      CALL DASH
      CALL CURVE(XX,SECTI,NPOINTS,0)
      CALL RESET('DASH')
      NPOINTS=0
C
C - JUMP BACK INTO LOOP FOR NEXT STRING OF PHASE DATA
      IF (KINCR.LT.NT) GO TO 437
  440 CONTINUE
      CALL ENDPL(0)
      GO TO 490
  450 CONTINUE
C
C -- VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); PHASE
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL INTAXS
      CALL MX1ALF('STANDARD','!')
      CALL MX2ALF('L/CGRK','#')
      CALL AREA2D(GRFSZ,GRFSZ)
      IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
         CALL MESSAG('SOLID = INTERFER.$',100,0.1,7.35)
         CALL MESSAG('DASHED = ACTUAL$',100,2.4,7.35)
      ENDIF
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO (490,452,490, 490,453,490, 490,454,490,
     1       490,455,490, 490,456,490, 490,457,490) MSRF
  452 CALL HEADIN('RAMAN PUMP: PHASE$',100,1.5,1)
      GO TO 459
  453 CALL HEADIN('RAMAN PUMP: PHASE (FFT)$',100,1.5,1)
      GO TO 459
  454 CALL HEADIN('RAMAN STOKES: PHASE$',100,1.5,1)
      GO TO 459
  455 CALL HEADIN('RAMAN STOKES: PHASE (FFT)$',100,1.5,1)
      GO TO 459
  456 CALL HEADIN('RAMAN MAT. EXC.: PHASE$',100,1.5,1)
      GO TO 459
  457 CALL HEADIN('RAMAN MAT. EXC.: PHASE (FFT)$',100,1.5,1)
  459 CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) THEN
         CALL XNONUM
         CALL XTICKS(0)
         CALL YNAME('FFT PHASE (MULTIPLES OF #PI)$',100)
         RDYX=RDYF
         XORIG=YFORIG
         XSTP=YFSTP
         XMAX=YFMAX
      ELSE
         CALL XNAME('Y-DIMENSION (CM)$',100)
         CALL YNAME('PHASE (MULTIPLES OF #PI)$',100)
         RDYX=RDY
         XORIG=YORIG
         XSTP=YSTP
         XMAX=YMAX
      ENDIF
      IF (NT.LE.8) THEN
         CALL MESSAG('EXPON., NHYP = $',100,0.1,7.1)
         CALL INTNO(NHYP,2.0,7.1)
         CALL MESSAG('1 - D STA.$',100,5.9,7.35)
         ISEC=NINT(SECR)
         IF (NT.GT.1) THEN
            CALL MESSAG('CASE$',100,4.0,7.1)
            CALL INTNO(ISEC,4.7,7.1)
         ENDIF
      ELSE
         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
         CALL MESSAG('T =  $',100,4.2,7.1)
         IPLACE=2
         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
         CALL REALNO(SECR,IPLACE,4.7,7.1)
         ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
      ENDIF
C
C - MAGNITUDE DATA, ABSCISSA VECTOR
      DO 460 K1=1,NY
      XR=SRF(ISEC,K1)
      XI=SRFI(ISEC,K1)
      SECTN(K1)=SQRT(XR*XR+XI*XI)
      XX(K1)=XORIG+RDYX*(K1-1)
 460  CONTINUE
C
C - UNCERTAIN PHASE THRESHOLD
      XMX=SECTN(ISMAX(NY,SECTN,1))
      IF (XMX.LT.1.0E-30) THEN
         WRITE (59,*) 'note: UNRELIABLE PHASE, MAGNITUDES ARE ZERO'
         GO TO 490
      ENDIF
      XTHRSH=XMX/1.0E8
C
C -- CALCULATE PHASE DATA
C
C - PHASE OF FIRST DATA POINT WITHIN +/- PI OF ZERO PHASE
      SCTOLD=0.0
C
C - INITIALIZE LOOP VARIABLES
      NAB=1
      NAN=0
      KINCR=0
C
C -- LOOP OVER ALL GRID POINTS; KINCR LOOP COUNTER
 470  CONTINUE
      KINCR=KINCR+1
C
C - CLEAR VECTOR FOR PHASE DATA
      SECTI(KINCR)=0.0
C
C - FIND STRING OF GRID POINTS (NAB TO NAN) WHERE MAGNITUDE OF FIELD
C   DATA EXCEEDS THRESHOLD
      IF (SECTN(KINCR).GE.XTHRSH) THEN
         NAN=KINCR
```

```
C
C - PLACE MARKER (SECTN=-1.0) WHERE FIELD MAGNITUDE IS BELOW THRESHOLD
C   (UNCERTAIN PHASE INFORMATION)
        ELSE
          SECTN(KINCR)=-1.0
          IF (NAN.GE.NAB) THEN
C
C - EXIT LOOP TEMPORARILY TO CALCULATE PHASE DATA FOR STRING OF GRID
C   POINTS NAB TO NAN
            GO TO 471
          ELSE
C
C - CURRENT DATA POINT STILL UNCERTAIN; INCREMENT NAB (BEGINNING OF NEXT
C   STRING)
            NAB=KINCR+1
          ENDIF
        ENDIF
C
C -- END LOOP OR CONTINUE IN LOOP UNTIL NT
      IF (KINCR.LT.NY) GO TO 470
C
C - SKIP PHASE CALCULATION, LAST DATA POINTS ARE UNCERTAIN
      IF (NAN.LT.NAB) GO TO 479
  471 CONTINUE
C
C - CALCULATE RAW PHASE MODULO 2*PI
      DO 473 K3=NAB,NAN
      SECTI(K3)=ATAN2(SRFI(ISEC,K3),SRF(ISEC,K3))/PI
  473 CONTINUE
C
C - CALCULATE EXACT PHASE KEEPING TRACK OF MULIPLES OF 2*PI COUNTED BY
C   LPI
      LPI=0
      IF (NAN.EQ.NAB) THEN
C
C - SINGLE DATA POINT
        SECTI(NAN)=SECTI(NAN)+SCTOLD
        GO TO 477
      ENDIF
C
C - PHASE OF FIRST DATA POINT IN STRING
      PSIP=SECTI(NAB)
      SECTI(NAB)=PSIP+SCTOLD
C
C - PHASE OF FOLLOWING DATA POINTS IN STRING
      DO 475 K4=NAB+1,NAN
      PSIK=SECTI(K4)
      IF (PSIP.GE.0.0) THEN
C
C - INCREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
C   THE PREVIOUS POINT PSIK (WHICH WAS POSITIVE)
        IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI+2
      ELSE
C
C - DECREMENT LPI IF PRESENT RAW PHASE PSIK DIFFERS BY MORE THAN PI FROM
C   THE PREVIOUS POINT PSIK (WHICH WAS POSITIVE)
        IF (ABS(PSIK-PSIP).GT.1.0) LPI=LPI-2
      ENDIF
C
C - EXACT PHASE
      SECTI(K4)=PSIK+LPI+SCTOLD
C
C - CURRENT RAW PHASE BECOMES PREVIOUS RAW PHASE NEXT TIME THROUGH THE
```

```
      C   LOOP
            PSIP=PSIK
        475 CONTINUE
        477 CONTINUE
      C
      C - STORE PHASE OF LAST DATA POINT AS REFERENCE VALUE FOR NEXT STRING OF
      C   RELIABLE DATA
            SCTOLD=SECTI(NAN)
      C
      C - INCREMENT LABEL OF BEGINNING OF NEXT STRING
            NAB=KINCR+1
      C
      C - FIND NEXT STRING OF PHASE DATA
            IF (NAN.LT.NY) GO TO 470
        479 CONTINUE
      C
      C - MEMORIZE ORIGINAL PHASE
            IF (ZVAL.LT.ZSTEP) THEN
               IF (MSRF.EQ.2) THEN
                  DO 480 I3=1,NY
                  PPMEM(I3,K)=SECTI(I3)
        480       CONTINUE
               ELSE
                  DO 481 I3=1,NY
                  PSMEM(I3,K)=SECTI(I3)
        481       CONTINUE
               ENDIF
            ELSE
      C
      C - INTERFEROMETRIC PHASE (CURRENT PHASE MINUS ORIGINAL PHASE)
               IF (MSRF.EQ.2) THEN
                  DO 482 I3=1,NY
                  SECTJ(I3)=SECTI(I3)-PPMEM(I3,K)
        482       CONTINUE
               ENDIF
               IF (MSRF.EQ.8) THEN
                  DO 483 I3=1,NY
                  SECTJ(I3)=SECTI(I3)-PSMEM(I3,K)
        483       CONTINUE
               ENDIF
            ENDIF
      C
      C -- COMPUTE COORDINATE SYSTEM
      C
      C - SCALE Y-COORDINATE AXIS
            NECLEC=1
            PHASTP=0.0
            CALL NYSXIS(SECTI,NY,NECLEC,PHASOR,PHASTP,PHASMX)
      C
      C - INCLUDE INTERFERENCE PHASE IN SCALE OF Y-AXIS LIMITS
            IF (MSRF.EQ.2.OR.MSRF.EQ.8) THEN
               NECLEC=0
               CALL NYSXIS(SECTJ,NY,NECLEC,PHASOR,PHASTP,PHASMX)
            ENDIF
      C
      C - MAKE FRACTIONAL Y-AXIS LIMITS INTEGRAL
            IF (PHASTP.LE.1.0) THEN
               PHSORI=ANINT(PHASOR)
               PHSMXI=ANINT(PHASMX)
               IF (PHASOR.LE.PHSORI) THEN
                  PHASOR=PHSORI-1.0
               ELSE
                  PHASOR=PHSORI
```

```
         ENDIF
         IF (PHASMX.GE.PHSMXI) THEN
            PHASMX=PHSMXI+1.0
         ELSE
            PHASMX=PHSMXI
         ENDIF
C
C - SMALLEST Y-INTERVAL SIZES SHOULD BE PI/4 AND PI/2
         PHASDF=PHASMX-PHASOR
         IF (PHASDF.LE.2.0) THEN
            PHASTP=0.25
         ELSE IF (PHASDF.LE.4.0) THEN
            PHASTP=0.5
         ENDIF
      ENDIF
C
C -- PLOT COORDINATE SYSTEM
      CALL GRAF(XORIG,XSTP,XMAX,PHASOR,PHASTP,PHASMX)
C
C - COMPLETE COORDINATE SYSTEM BY A FRAME AND TICKMARKS
      XDUM(1)=XORIG
      XDUM(2)=XMAX
      YDUM(1)=PHASMX
      YDUM(2)=PHASMX
      CALL CURVE(XDUM,YDUM,2,0)
      IF (MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17) GOTO 485
      NTIK=NINT((XMAX-XORIG)/XSTP)
      YDUM(1)=PHASMX
      YDUM(2)=PHASMX-(PHASMX-PHASOR)/50.0
      XDM=XORIG
      DO 484 ITK=1,NTIK-1
      XDM=XDM+XSTP
      XDUM(1)=XDM
      XDUM(2)=XDM
      CALL CURVE(XDUM,YDUM,2,0)
 484  CONTINUE
 485  CONTINUE
      XDUM(1)=XMAX
      XDUM(2)=XMAX
      YDUM(1)=PHASMX
      YDUM(2)=PHASOR
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((PHASMX-PHASOR)/PHASTP)
      XDUM(1)=XMAX-(XMAX-XORIG)/50.0
      XDUM(2)=XMAX
      YDM=PHASOR
      DO 486 ITK=1,NTIK-1
      YDM=YDM+PHASTP
      YDUM(1)=YDM
      YDUM(2)=YDM
      CALL CURVE(XDUM,YDUM,2,0)
 486  CONTINUE
C
C - PLOT PHASE CURVE SEGMENTS; RESET COUNTERS
      NPOINTS=0
      KINCR=0
C
C -- LOOP OVER DATA POINTS; LOOP COUNTER KINCR
 487  CONTINUE
      KINCR=KINCR+1
      IF (SECTN(KINCR).LT.-0.99) THEN
C
C - UNRELIABLE PHASE MARKER ENCOUNTERED; PLOT DATA STRING OF LENGTH
```

```
C   NPOINTS
            IF (NPOINTS.GT.0) GO TO 488
         ELSE
C
C - INCREMENT DATA STRING COUNTER; PUSH RELIABLE DATA TO FRONT OF VECTOR
            NPOINTS=NPOINTS+1
            SECTI(NPOINTS)=SECTI(KINCR)
            IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
               SECTJ(NPOINTS)=SECTJ(KINCR)
            ENDIF
            XX(NPOINTS)=XX(KINCR)
         ENDIF
C
C -- END LOOP OR CONTINUE IN LOOP UNTIL NY
         IF (KINCR.LT.NY) GO TO 487
C
C - LAST DATA POINTS UNRELIABLE; END PHASE PLOTTING
         IF (NPOINTS.EQ.0) GO TO 489
  488    CONTINUE
C
C - PLOT DATA STRING; RESET DATA COUNTER
C - DRAW INTERFEROMETRIC PHASE SOLID
         IF (ZVAL.GE.ZSTEP.AND.(MSRF.EQ.2.OR.MSRF.EQ.8)) THEN
            CALL CURVE(XX,SECTJ,NPOINTS,0)
         ENDIF
C
C - DRAW CURRENT PHASE DASHED
         CALL DASH
         CALL CURVE(XX,SECTI,NPOINTS,0)
         CALL RESET('DASH')
         NPOINTS=0
C
C - JUMP BACK INTO LOOP FOR NEXT STRING OF PHASE DATA
         IF (KINCR.LT.NY) GO TO 487
  489    CONTINUE
C
C -- SPECIAL AXIS AND LABEL FOR FFT COORDINATE
C
         IF (NY.GT.8.AND.(MSRF.EQ.5.OR.MSRF.EQ.11.OR.MSRF.EQ.17))
       1    CALL XISFFT('X',PHASOR,PHASMX)
C
C - END OF PHASE SECTION PLOT
         CALL ENDPL(0)
  490    CONTINUE
C
C --- CROSS SECTIONS OF AMPLITUDE SURFACES (REAL/IMAGINARY
C        REPRESENTATION)
         NSRF=MSRF+1
C
C -- CHECK EACH ELEMENT IN ROW MSRF OF ARRAY CSEC
         DO 590 K=1,NSEC
         SECR=REAL(CSEC(NSRF,K))
C
C - ONE-DIMENSIONAL CASES
         IF (NY.LE.8) THEN
            IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 590
            GO TO 501
         ELSE IF (NT.LE.8) THEN
            IF (SECR.LT.0.5.OR.SECR.GT.8.5) GO TO 590
            GO TO 540
         ENDIF
C
C - TWO DIMENSIONAL CASES; SECTION ONLY IF IMAGINARY PART OF CSEC-
```

```
C   ELEMENT IS EQUAL TO 1.0 OR 2.0;
C   OTHERWISE GO TO NEXT LOOP COUNTER VALUE K, I.E. NEXT ELEMENT OF LINE
C   NSRF IN ARRAY CSEC
      SECI=AIMAG(CSEC(NSRF,K))
      IF (SECI.GT.0.9.AND.SECI.LT.1.1) GO TO 540
      IF (SECI.GT.1.9.AND.SECI.LT.2.1) GO TO 501
      GO TO 590
 501  CONTINUE
C
C -- HORIZONTAL CROSS SECTION (SECOND ARGUMENT OF SRF FIXED); AMPLITUDE
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL AREA2D(GRFSZ,GRFSZ)
      CALL INTAXS
      CALL MESSAG('SOLID = REAL$',100,0.1,7.35)
      CALL MESSAG('DASHED = IMAG.$',100,1.7,7.35)
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO(590,590,502, 590,590,503, 590,590,504,
     1     590,590,505, 590,590,506, 590,590,507) NSRF
 502  CALL HEADIN('RAMAN PUMP: AMPLITUDE$',100,1.5,1)
      GO TO 509
 503  CALL HEADIN('RAMAN PUMP: MODE AMPLITUDE$',100,1.5,1)
      GO TO 509
 504  CALL HEADIN('RAMAN STOKES: AMPLITUDE$',100,1.5,1)
      GO TO 509
 505  CALL HEADIN('RAMAN STOKES: MODE AMPLITUDE$',100,1.5,1)
      GO TO 509
 506  CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE$',100,1.5,1)
      GO TO 509
 507  CALL HEADIN('RAMAN MAT. EXC.: MODE AMPLITUDE$',100,1.5,1)
 509  CONTINUE
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      CALL XNAME('TIME (PICO-SECONDS)$',100)
      IF (NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18) THEN
         CALL YNAME('MODE AMPLITUDE$',100)
         CALL MESSAG('KY =  $',100,4.2,7.1)
         IPLACE=2
         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
         CALL REALNO(SECR,IPLACE,4.7,7.1)
         ISEC=INT((SECR+NYHP/YM2M1)*YM2M1)
      ELSE
         CALL YNAME('AMPLITUDE$',100)
         IF (NY.LE.8) THEN
            CALL MESSAG('1 - D TRA.$',100,5.9,7.35)
            ISEC=NINT(SECR)
            GO TO (512,513,514,515) ITYPE(ISEC)
 512        CALL MESSAG('SEC-HYPERB. , EXP = $',100,0.1,7.1)
            GO TO 516
 513        CALL MESSAG('RECTANGULAR$',100,0.1,7.1)
            GO TO 516
 514        CALL MESSAG('LORENTZIAN  , EXP = $',100,0.1,7.1)
            GO TO 516
 515        CALL MESSAG('EXPONENTIAL , EXP = $',100,0.1,7.1)
 516        CONTINUE
            IF (ITYPE(ISEC).NE.2) THEN
               XRTYPE=RTYPE(ISEC)
               IPLACE=2
               IF (ABS(XRTYPE).GT.9999.0.OR.ABS(XRTYPE).LT.0.01)
     1         IPLACE=-2
               CALL REALNO(XRTYPE,IPLACE,2.4,7.1)
            ENDIF
            IF (NY.GT.1) THEN
               CALL MESSAG('CASE$',100,4.0,7.1)
               CALL INTNO(ISEC,4.7,7.1)
            ENDIF
         ELSE
            CALL MESSAG('Y =  $',100,4.2,7.1)
            IPLACE=2
            IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
            CALL REALNO(SECR,IPLACE,4.7,7.1)
            CALL MESSAG('2 - DIM.$',100,5.9,7.35)
            ISEC=INT((SECR+RDY*NYHP)/RDY)
         ENDIF
      ENDIF
C
C - CROSS SECTION DATA
      DO 520 K1=1,NT
      SECTN(K1)=SRF(K1,ISEC)
      SECTI(K1)=SRFI(K1,ISEC)
      XX(K1)=TM1+RDT*(K1-1)
 520  CONTINUE
C
C - DRAW COORDINATE SYSTEM
      NECLEC=1
      WSTP=0.0
      CALL NYSXIS(SECTN,NT,NECLEC,WORIG,WSTP,WMAX)
      NECLEC=0
      WSTP=0.0
      CALL NYSXIS(SECTI,NT,NECLEC,WORIG,WSTP,WMAX)
      CALL GRAF(TORIG,TSTP,TMAX,WORIG,WSTP,WMAX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
      XDUM(1)=TORIG
      XDUM(2)=TMAX
      YDUM(1)=WMAX
      YDUM(2)=WMAX
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((TMAX-TORIG)/TSTP)
      YDUM(1)=WMAX
      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
      XDM=TORIG
      DO 536 ITK=1,NTIK-1
      XDM=XDM+TSTP
      XDUM(1)=XDM
      XDUM(2)=XDM
      CALL CURVE(XDUM,YDUM,2,0)
 536  CONTINUE
      XDUM(1)=TMAX
      XDUM(2)=TMAX
      YDUM(1)=WMAX
      YDUM(2)=WORIG
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((WMAX-WORIG)/WSTP)
      XDUM(1)=TMAX-(TMAX-TORIG)/50.0
      XDUM(2)=TMAX
      YDM=WORIG
      DO 537 ITK=1,NTIK-1
      YDM=YDM+WSTP
      YDUM(1)=YDM
      YDUM(2)=YDM
      CALL CURVE(XDUM,YDUM,2,0)
```

```
  537 CONTINUE
C
C - DRAW CROSS SECTION CURVES
      NPOINTS=NT
      CALL CURVE(XX,SECTN,NPOINTS,0)
      CALL DASH
      CALL CURVE(XX,SECTI,NPOINTS,0)
      CALL RESET('DASH')
      CALL ENDPL(0)
      GO TO 590
  540 CONTINUE
C
C -- VERTICAL CROSS SECTION ( FIRST VARIABLE FIXED IN SRF); AMPLITUDE
C
C - START A NEW GRAPHICS FRAME FOR THIS CROSS SECTION
      CALL RESET('ALL')
      CALL AREA2D(GRFSZ,GRFSZ)
      CALL INTAXS
C
C - HEADLINE, LABELS, AND PARAMETER
      GO TO (590,590,542, 590,590,543, 590,590,544,
     1       590,590,545, 590,590,546, 590,590,547) NSRF
  542 CALL HEADIN('RAMAN PUMP: AMPLITUDE$',100,1.5,1)
      GO TO 549
  543 CALL HEADIN('RAMAN PUMP: AMPLITUDE (FFT)$',100,1.5,1)
      GO TO 549
  544 CALL HEADIN('RAMAN STOKES: AMPLITUDE$',100,1.5,1)
      GO TO 549
  545 CALL HEADIN('RAMAN STOKES: AMPLITUDE (FFT)$',100,1.5,1)
      GO TO 549
  546 CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE$',100,1.5,1)
      GO TO 549
  547 CALL HEADIN('RAMAN MAT. EXC.: AMPLITUDE (FFT)$',100,1.5,1)
  549 CONTINUE
      CALL MESSAG('SOLID = REAL$',100,0.1,7.35)
      CALL MESSAG('DASHED = IMAG.$',100,1.7,7.35)
      CALL MESSAG('Z = $',100,5.9,7.1)
      IPLACE=2
      IF (ABS(ZVAL).GT.9999.0.OR.ABS(ZVAL).LT.0.01) IPLACE=-2
      CALL REALNO(ZVAL,IPLACE,6.4,7.1)
      IF (NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18) THEN
         CALL XNONUM
         CALL XTICKS(0)
         CALL YNAME('FFT AMPLITUDE$',100)
         RDYX=RDYF
         XORIG=YFORIG
         XSTP=YFSTP
         XMAX=YFMAX
      ELSE
         CALL XNAME('Y-DIMENSION (CM)$',100)
         CALL YNAME('AMPLITUDE$',100)
         RDYX=RDY
         XORIG=YORIG
         XSTP=YSTP
         XMAX=YMAX
      ENDIF
      IF (NT.LE.8) THEN
         CALL MESSAG('EXPON., NHYP = $',100,0.1,7.1)
         CALL INTNO(NHYP,2.0,7.1)
         CALL MESSAG('1 - D STA.$',100,5.9,7.35)
         ISEC=NINT(SECR)
         IF (NT.GT.1) THEN
            CALL MESSAG('CASE$',100,4.0,7.1)
            CALL INTNO(ISEC,4.7,7.1)
         ENDIF
      ELSE
         CALL MESSAG('T =  $',100,4.2,7.1)
         IPLACE=2
         IF (ABS(SECR).GT.9999.0.OR.ABS(SECR).LT.0.01) IPLACE=-2
         CALL REALNO(SECR,IPLACE,4.7,7.1)
         CALL MESSAG('2 - DIM.$',100,5.9,7.35)
         ISEC=INT((SECR+RDT*(NT/2+1))/RDT)
      ENDIF
C
C - CROSS SECTION DATA
      DO 555 K1=1,NY
      SECTN(K1)=SRF(ISEC,K1)
      SECTI(K1)=SRFI(ISEC,K1)
      XX(K1)=XORIG+RDYX*(K1-1)
 555  CONTINUE
C
C - DRAW COORDINATE SYSTEM
      NECLEC=1
      WSTP=0.0
      CALL NYSXIS(SECTN,NY,NECLEC,WORIG,WSTP,WMAX)
      NECLEC=0
      WSTP=0.0
      CALL NYSXIS(SECTI,NY,NECLEC,WORIG,WSTP,WMAX)
      CALL GRAF(XORIG,XSTP,XMAX,WORIG,WSTP,WMAX)
C
C - COMPLETE COORDINATE FRAME AND TICKMARKS
      XDUM(1)=XORIG
      XDUM(2)=XMAX
      YDUM(1)=WMAX
      YDUM(2)=WMAX
      CALL CURVE(XDUM,YDUM,2,0)
      IF (NY.GT.8.AND.(NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18)) GOTO 568
      NTIK=NINT((XMAX-XORIG)/XSTP)
      YDUM(1)=WMAX
      YDUM(2)=WMAX-(WMAX-WORIG)/50.0
      XDM=XORIG
      DO 567 ITK=1,NTIK-1
      XDM=XDM+XSTP
      XDUM(1)=XDM
      XDUM(2)=XDM
      CALL CURVE(XDUM,YDUM,2,0)
 567  CONTINUE
 568  CONTINUE
      XDUM(1)=XMAX
      XDUM(2)=XMAX
      YDUM(1)=WMAX
      YDUM(2)=WORIG
      CALL CURVE(XDUM,YDUM,2,0)
      NTIK=NINT((WMAX-WORIG)/WSTP)
      XDUM(1)=XMAX-(XMAX-XORIG)/50.0
      XDUM(2)=XMAX
      YDM=WORIG
      DO 569 ITK=1,NTIK-1
      YDM=YDM+WSTP
      YDUM(1)=YDM
      YDUM(2)=YDM
      CALL CURVE(XDUM,YDUM,2,0)
 569  CONTINUE
C
C - DRAW CROSS SECTION CURVES
      NPOINTS=NY
      CALL CURVE(XX,SECTN,NPOINTS,0)
      CALL DASH
      CALL CURVE(XX,SECTI,NPOINTS,0)
      CALL RESET('DASH')
C
C - AXIS AND LABEL FOR FFT COORDINATE
      IF (NY.GT.8.AND.(NSRF.EQ.6.OR.NSRF.EQ.12.OR.NSRF.EQ.18))
     1   CALL XISFFT('X',WORIG,WMAX)
C
C - END AMPLITUDE SECTIONS
      CALL ENDPL(0)
  590 CONTINUE
      RETURN
      END
c
c
c
c
      SUBROUTINE NYSXIS(VEC,NPOINTS,NECLEC,VECBOT,VECGAP,VECTOP)
c
C  This subroutine was written by Godehard Hilfer (3/87). It finds
C  'nice' end-values (vecbot,vectop) and intervals (vecgap) for
C  linear coordinate axes.
c
C  The subroutine can find such values around the extremas of the
C  argument veC and/or around the input values of the arguments vecbot
C  and vectop. This is determined by the argument necvec. If
C                necveC =-1 then the input vector veC is neglected and
C                           'nice' limits and interval are only based on
C                           the current values of vecbot and vectop. If
C                necveC = 0 then vecbot and vectop are also incorporated
C                           in the search for the extrema of vec,
C                           thereby allowing user controled lower
C                           limits for these extrema. If
C                necveC = 1 then current values of vecbot and vectop are
C                           neclected and 'nice' limits and interval
C                           based on the npoints values in veC alone.
C  It is also possible to 'hard-wire' the lower (upper) end-value to
C  the current value of vecbot (vectop) by setting the argument vecgap
C  to -1.0 (1.0) as input. If vectop=2.0 on input both end values are
C  'hard-wired'.
c
C  The subroutine finds the extrema of the input data. Then it
C  determines the largest integral power of ten (xtrpow) that is still
C  smaller than the larger of the absolute values of the extrema.
C  Based on xtrpow the leading two decimal places of the extrema are
C  compared with each other. The possible difference in the leading
C  decimal places the extrema belong to one of seven interval classes
C  with the following interval sizes: 0.005, 0.05, 0.1, 0.2, 0.5, 1.0,
C  2.0 times xtrpow. The extremal values are one interval beyond the
C  integer that is closest to the extrema. If the hard-wiring option
C  was chosen the hard-wired end value is reinstated before the
C  interval and end values are returned to the calling routine.
c
C                          --variables--
c
C     mantdif = difference in integral mantissa of extrema
C     mantlw = lower extremum integral mantissa
C     mantup = upper extremum integral mantissa
C     nechrd = hard-wiring flag
C     necveC = flag that picks input data
C     npoints = number of elements to be considered in data vector vec
C     vcevnl = even lower extremum guide
```

```
C        vcevnu = even upper extremum guide
C        vchdbt = hard-wired bottom value
C        vchdtp = hard-wired top value
C        vcmntl = lower extremum divided by dominant power of 10
C        vcmntu = upper extremum divided by dominant power of 10
C        veC = data vector
C        vecbot = data minimum and returns 'nice' lower value
C        vecgap = hard wiring flag on input; 'nice' interval on output
C        vecmax = upper data extremum
C        vecmin = lower data extremum
C        vectop = data maximum and returns 'nice' upper value
C        xtrpow = dominant power of 10
C        xtrpwl = next integral power of 10 below lower extremum
C        xtrpwu = next integral power of 10 below upper extremum
C -----------------------------------------------------------------
c
      PARAMETER (NT=256,NY=128,NTPY=NT+NY)
C
      DIMENSION VEC(NTPY)
C
C - STORE INPUT VALUES
      VCHDBT=VECBOT
      VCHDTP=VECTOP
      NECHRD=NINT(100.0*VECGAP)
C
C - CORRECT OR RETURN UPON ERRONEUS INPUT
      IF (NECHRD.NE.-100.AND.NECHRD.NE.100.AND.NECHRD.NE.200) NECHRD=0
      IF (NECLEC.NE.-1.AND.NECLEC.NE.0.AND.NECLEC.NE.1) THEN
         WRITE (59,*) 'note: NECLEC IN SUBROUTINE NYSXIS OUT OF RANGE'
         RETURN
      ENDIF
      IF (NECLEC.LT.1.AND.VECBOT.GE.VECTOP) THEN
         WRITE (59,*) 'note: VECBOT IS GREATER THAN OR EQUAL TO VECTOP
     1         IN NYSXIS'
         VECBOT=AMIN1(VECBOT,VECTOP)
         VECTOP=AMAX1(VECBOT,VECTOP)
      ENDIF
C
C - FIND EXTREMA
      NECLEC=NECLEC+2
      GO TO (810,820,830) NECLEC
  810 CONTINUE
      VECMIN=VECBOT
      VECMAX=VECTOP
      GO TO 840
  820 CONTINUE
      VECMIN=VEC(ISMIN(NPOINTS,VEC,1))
      IF (NECHRD.EQ.-100.OR.NECHRD.EQ.200.AND.VECMIN.LT.VECBOT) THEN
         WRITE (59,*) 'warning: FUNCTION EXTENDS BELOW AXIS'
      ENDIF
      VECMAX=VEC(ISMAX(NPOINTS,VEC,1))
      IF (NECHRD.EQ.100.OR.NECHRD.EQ.200.AND.VECMAX.GT.VECTOP) THEN
         WRITE (59,*) 'warning: FUNCTION EXTENDS ABOVE AXIS'
      ENDIF
      VECMIN=AMIN1(VECMIN,VECBOT)
      VECMAX=AMAX1(VECMAX,VECTOP)
      GO TO 840
  830 CONTINUE
      VECMIN=VEC(ISMIN(NPOINTS,VEC,1))
      VECMAX=VEC(ISMAX(NPOINTS,VEC,1))
  840 CONTINUE
C
C - CONSIDER HARDWIRED VALUES AS EXTREMA
```

```
      IF (NECHRD.EQ.-100) THEN
         VECMIN=AMIN1(VECBOT,VECMIN)
      ELSE IF (NECHRD.EQ.100) THEN
         VECMAX=AMAX1(VECTOP,VECMAX)
      ELSE IF (NECHRD.EQ.200) THEN
         VECMIN=AMIN1(VECBOT,VECMIN)
         VECMAX=AMAX1(VECTOP,VECMAX)
      ENDIF
C
C - FIND DOMINANT INTEGRAL POWER OF TEN FOR THE EXTREMA
      RCUT=1.0E-35
      IF (ABS(VECMAX).GT.RCUT) THEN
         CALL POWBAS(VECMAX,XTRPWU)
         IF (ABS(VECMIN).GT.RCUT) THEN
            CALL POWBAS(VECMIN,XTRPWL)
            XTRPOW=MAX(XTRPWU,XTRPWL)
         ELSE
            XTRPOW=XTRPWU
         ENDIF
      ELSE
         CALL POWBAS(VECMIN,XTRPOW)
      ENDIF
C
C - FIND MANTISSA OF THE EXTREMA
      VCMNTU=VECMAX/XTRPOW
      VCMNTL=VECMIN/XTRPOW
C
C - CONSTANTS OR EXTREMA THAT DIFFER BY LESS THAN ONE IN THE
C   THIRD SIGNIFICANT PLACE
      IF (ABS(VCMNTU-VCMNTL).LE.0.01) THEN
         VCEVNU=0.01*(NINT(100.0*VCMNTU)+1)
         VCEVNL=0.01*(NINT(100.0*VCMNTL)-1)
         VECGAP=0.005*XTRPOW
         GO TO 880
      ENDIF
C
C - MAKE INTEGER OUT OF THE LEADING TWO SIGNIFICANT PLACES
      MANTUP=NINT(10.0*VCMNTU)
      MANTLW=NINT(10.0*VCMNTL)
      MANTDIF=ABS(MANTUP-MANTLW)
C
C - EXTREMA DIFFER BY LESS THAN 2 PERCENT
      IF (MANTDIF.LT.2) THEN
         VCEVNU=0.05*(INT(NINT(100.0*VCMNTU)/5)+1)
         VCEVNL=0.05*(INT(NINT(100.0*VCMNTL)/5)-1)
         VECGAP=0.05*XTRPOW
C
C - EXTREMA DIFFER BY LESS THAN 10 PERCENT
      ELSE IF (MANTDIF.LT.10) THEN
         VCEVNU=0.1*(MANTUP+1)
         VCEVNL=0.1*(MANTLW-1)
         VECGAP=0.1*XTRPOW
C
C - EXTREMA DIFFER BY LESS THAN 20 PERCENT
      ELSE IF (MANTDIF.LT.20) THEN
         VCEVNU=0.2*(INT(MANTUP/2)+1)
         VCEVNL=0.2*(INT(MANTLW/2)-1)
         VECGAP=0.2*XTRPOW
C
C - EXTREMA DIFFER BY LESS THAN 50 PERCENT
      ELSE IF (MANTDIF.LT.50) THEN
         VCEVNU=0.5*(INT(MANTUP/5)+1)
         VCEVNL=0.5*(INT(MANTLW/5)-1)
         VECGAP=0.5*XTRPOW
C
C - EXTREMA DIFFER BY LESS THAN 100 PERCENT
      ELSE IF (MANTDIF.LT.100) THEN
         VCEVNU=1.0*(INT(MANTUP/10)+1)
         VCEVNL=1.0*(INT(MANTLW/10)-1)
         VECGAP=XTRPOW
C
C - EXTREMA DIFFER BY MORE THAN 100 PERCENT (E.G. OPPOSITE SIGN)
      ELSE
         VCEVNU=2.0*(INT(MANTUP/20)+1)
         VCEVNL=2.0*(INT(MANTLW/20)-1)
         VECGAP=2.0*XTRPOW
      ENDIF
 880  CONTINUE
C
C - HARD-WIRED LOWER END VALUE
      IF (NECHRD.EQ.-100) THEN
         VECTOP=VCEVNU*XTRPOW
C
C - NO HARD-WIRED END VALUE
      ELSE IF (NECHRD.EQ.0) THEN
         VECTOP=VCEVNU*XTRPOW
         VECBOT=VCEVNL*XTRPOW
C
C - HARD-WIRED UPPER END VALUE
      ELSE IF (NECHRD.EQ.100) THEN
         VECBOT=VCEVNL*XTRPOW
      ENDIF
      RETURN
      END
c
c
c
c
      SUBROUTINE POWBAS(VARBLE,PWDECN)
c
C  This subroutine was written by Godehard Hilfer (3/87). It determines
C  the next lower integral power of 10, pwdecn, of the quantity varble.
C  If varble vanishes pwdecn returns 1.0.
c
      RCUT=1.0E-35
      VABS=ABS(VARBLE)
      IF (VABS.GT.RCUT) GO TO 10
      PWDECN=1.0E-36
      RETURN
 10   CONTINUE
      XPLOG=ALOG10(VABS)
      PWDECN=10.0**INT(XPLOG)
      IF (XPLOG.LT.0.0) PWDECN=PWDECN/10.0
      RETURN
      END
```

APPENDIX B

Manual

MANUAL

RAMAN AMPLIFIER CODE RAM2D1
AND
ASSOCIATED DIAGNOSTIC PROGRAM PRAM1

INTRODUCTION

The manual at hand is intended to introduce the reader to the use of the (2+1)-dimensional Raman amplifier code RAM2D1 and the accompanying diagnostic program PRAM1 as installed on the CRAY X-MP 24[1] computer of the Central Computing Facility (CCF) of the U.S. Naval Research Laboratory (NRL).

Both programs are written in CRAY-FORTRAN (CFT) and run under the CRAY operating system (COS). The computational setup at NRL favors batch job operation. In this mode, the user does not interact directly with the CRAY computer while working with RAM2D1 or PRAM1. Four Digital Equipment Corporation (DEC) VAX[2] computers (called NRL1, NRL2, NRL3, NRL4) process independently and simultaneously the requests of all users for communication, editing, storage, etc. Any of the four machines can be used interchangeably. Due to size and speed requirements most computations when using RAM2D1 and PRAM1 are done on the CRAY computer. Presently the only computational use of the VAXes is the post processing of the graphics data files that are generated by PRAM1. These data files contain device independent graphics data which he VAX software converts into data that can be displayed on a VT240-type terminal or a laser printer. All other computing is done on the CRAY.

Data storage is available4 separately both on the CRAY and on the VAX computers. Both primarily utilize quickly accessable hard disk storage devices. However, both locations offer also the more economical long term tape storage option. All files (datasets) in memory during computation on the CRAY computer are volatile. That means that a computational process has to be given explicitly all the necessary datasets and the results have to be retrieved explicitly from it; otherwise,l the datasets disappear upon completion of the job. The resulting data can be sent to the VAX for storage or can be stored on devices that are reserved for CRAY use only.

The data files resulting from the execution of RAM2D1 are programmed to be stored on the CRAY tape storage device (= off-line; CRAY disk = on- line). The code PRAM1 uses these data to produce a DISSPLA-META[3] file which is the device independent data file mentioned above. A batch job command transfers this file to the VAX computer post morten of PRAM1 for storage and/or post-processing. Through the VAX, the data can be displayed or printed.

The remainder of this manual contains explicit instructions and examples pertaining to the use of the computers and the programs RAM2d1 and PRAM1 so that the user can, with a particular input parameter choice in hand, run the codes and carry the results home on paper.

1 CRAY X-MP (and other CRAY logos) is a registered trademark of CRAY Research, Incorporated, Mendota Heights, MN.

2 VAX, DEC, and others are registered trademarks of the Digital Equipment Corporation, Maynard, MA.

3 DISSPLA is a registered trademark of the Integrated Software Systems Corporation, San Diego, CA.

CHAPTER I

GETTING STARTED

PART 1.A COMPUTER ACCESS / LOGIN

Section I.A.1: Telephone Access

For remote access, by means of a personal computer and the telephone network, find appropriate communications software (e.g. VTEK, KERMIT, or other) to dial Washington, D.C., metropolitan area phone number 767-2000 for a 1200 baud connection to the Naval Research Laboratory Central Computing Facility (NRL CCF). For a 2400 baud connection dial the number 767- 1240.

Should you have problems call 767-3512 for a status information on the CCF, or call the consultants desk 767-3542 for assistance Mondays through Fridays from 9am to 5pm.

```
After the connection is made type:        <     (carriage return)
                                          <
                                          .
                                          .
                                          .
```

two or more carriage returns until the computer prompts: # From here proceed to section I.A.4: DEC-server.

Section I.A.2: Hardwired Terminal Access

Access through one of the terminals at the CCF is obtained in the following way: Turn the power on.

```
type:      <        (return)
prompts:   You may now enter Net/One commands
           >
           >
type:       c cts <
prompts:   connecting ...  (1) ------ success
           #
```

From here proceed to section I.A.4: DEC-server.

Section I.A.3: Building A68 Access

The terminals in building A68 at NRL (John Reintjes' section) are connected to the communications server CS/200T which in turn is hardwired directly to the front-end VAX computers. Thereby the DEC-server involved in all other access paths is circumvented. Proceed as follows: Turn the power on.

```
type:      < (return)
prompts:   CS/200T>
type:      c nrl <
```

which will establish connection to NRL3 (alternatively type: `c nrl1 <`, or `c nrl2 <`, or `c nrl3 <`). From here proceed to section I.A.5: VAX login. In building A68 NRL4 can only be accessed through the DEC-server. For that, turn the power switch of the terminal to ON

```
type:      < (return)
prompts:   CS/200T>
type:      c lat-gw <
prompts:   Querying Primary Name Server...
           Connecting...  session 1 -- connected to lat-gw
type:      <
prompts:   Local>
```

which indicates successful access to the DEC-server. From here continue with what follows the prompt `Local>` in section I.A.4: DEC-server below.

Section I.A.4: Dec-Server

When the computer

prompts: #

type: `n` < (Note: the letter n will not show up on the screen!)

prompts: `Enter username>`

type: `'your username'` < (=name under which you may use the VAXes)

prompts: `Local>`

Now you have accessed the so-called DEC-server. (type: `help` < if you wish on-line information about this networking facility; otherwise:)

type: `c nrl` <

to be connected with one of the four VAX front-end computers. You may explicitly specify the VAX or your choice (e.g. `c nrl3` < , to get onto the NRL3 computer etc.). A standard VAX-login ensues. Proceed to section I.A.5: VAX Login.

Section I.A.5: VAX Login

The last pressing of the return key should effect that the system

prompts: `Username:`

whereupon it is necessary to

type: `'your username'` <

prompts: `Password:`

type: `'your password'` < (will not echo)

Then the VAX computer executes the login which will be finished when a $-sign appears on a line by itself following all other text on the screen.

From here proceed to Section I.B.2: Obtaining the Necessary Files, if you do not have them; or continue with Section I.B.3: Editing Files, if you do have all the files but need to change something; or turn to Chapter II, if you have the correct set of files for the intended simulation, or, if the simulation was previously done and the results need to be converted into graphs; or Chapter III, Viewing the Results, to see the graphs actually come out on the terminal screen or on paper; or turn to Part IV.A., File Storage, if programs or data need to be moved into storage or removed from it.

Section I.A.6: VAX Logout

To leave the computer system one has to logout. This procedure terminates all access to and responses from the computer. To logout

type: `log <`

which, e.g.,

prompts:
```
USER      logged out at d-a-t-e t:i:m:e
Local - Session x disconnected from NRL
Local>
```

This is the prompt of the DEC-server network node. A second LOG is necessary to signoff from it and to free the port of access. Thus,

type: `log <`

which will be acknowledged by telling from which port was logged off. In all it takes two log to finish the computing session. The second log is not necessary but neither harmful when using the building A68 communication server CS/200T. After that the terminal is disconnected from the CCF.

If the front-end VAXes do not obtain new input from the terminal within roughly 8 minutes, a ten minute countdown associated with two warnings, 5 minutes apart, ensues followed by an automatic logout of the inactive user.

Section I.A.7: CRAY Login/Logout

Access of the batch job to the CRAY is authorized by means of the first two batch job file command lines: 'JOB,---.' and 'ACCOUNT,---.' (see also subsection I.B.5.1: The Batch Job Command File). Access to the CRAY computer and its storage facilities is limited to the command lines in the batch job command file.

PART I.B FILE MANAGEMENT

Section I.B.1: Names of the Game

I.B.1.1 Code File Names in VAX/VMS

The nomenclature of all relevant files is as follows. The names of the source codes as indicated above, are RAM2D1 and PRAM1. The name RAM-2D- 1 abbreviates that this code solves the Raman amplification problem in 2-D. *i.e.*, the two spatial dimensions: z (linear coordinate along the central Stokes beam ray path) and y (linear coordinate orthogonally transverse to z). The character 1 in the name indicates that this is the first generation of this code. The diagnostic program name P-RAM-1 abbreviates: plots of the Raman amplifier code, 1st generation.

Both source codes reside on the VAX computers. Following the VAX/VMS operating system particulars, their full VAX-file names are:

DUA107:[HILFER.FOR]RAM2D1C.FOR;1
DUA107:[HILFER.FOR]PRAM1CD.FOR;1.

According to VAX/VMS conventions the name elements and their meanings are: DUA107: indicates the specific storage disk name on which the file is stored. [HILFER.FOR] indicates that the file belongs to the subdirectory FOR of the HILFER directory of files on that disk. The code name RAM2D1 was supplemented by suffix C to indicate version C of the code (see Appendix D for details and other versions). The arbitrarily chosen file extension .FOR is a reminder that the file contains a FORTRAN code. The file version ;1 is a number that serves the VAX computer to distinguish files of the same name. Every time the file is amended or changed, the VAX computer will keep the old file with its full old name and will create a new file with amendments and/or changes that will be given the same name but a version number one greater than that of the old file. Therefore, the highest version number indicates the most up-to-date version of the same file. The characters CD in PRAM1CD indicate that this version of PRAM1 works with RAM2D1C and RAM2D1D (i.e. on the NRL-CRAY, as opposed to PRAM1AB, which works with RAM2D1A and RAM2D1B on the NMFECC-CRAYs).

All relevant files are stored by default in the same directory on the same disk. Hence, the file name portion DUA107:[HILFER.FOR] is the same for all files and will be dropped in this manual for brevity's sake. Since the version number may be larger than 1 depending on, and only significant for, code development, the user can neglect it and it will be dropped also. Thus, the code names reduce to, simply,

RAM2D1C.FOR
PRAM1CD.FOR

I.B.1.2 Relevant Groups of Files

The relevant files can be grouped by their file name extensions (*i.e.*, three characters following the dot in the full file name analogous to what was described in subsection I.B.1.1: Code File Names in VAX/VMS). There are the following groups:

.FOR	(the 2 source code files mentioned in subsection I.B.1.1)
.DAT	(input and output data files)
.JOB	(batch job command files containing the user's commands for the CRAY computer)
.CPR	(message files generated by the CRAY computer system during job execution containing listings, messages, and a batch job log)
.MSG	(message files generated by the FORTRAN code during job execution containing formated and unformated output as programmed by the code developers.)
.TMP	(device specific graphics data files that can be printed on the laser printer)

I.B.1.3 Modes of Operation and Encryption of Dimensions

The code's operation as a two-dimensional or one-dimensional model is switched by the field array dimension parameters `NT` and `NY`. If both integers are larger than eight, two-dimensional operation is indicated and the algorithm expects that the parameters are set to integral powers of 2. If one of the parameters is 8 or smaller, the variable (t or y) associated with that parameter ceases to be a variable, and refers instead to the number of cases being run in the one dimesional mode. Both `NT` and `NY` must never be 8 or less simultaneously.

In short, the values of `NT` and `NY` are salient characteristics of any simulation and serve, therefore, to distinguish data files and code versions by contributing two characters to every file name. The first of both characters indicates the value of `NT`, the second that of `NY` according to the following scheme. If the value is 8 or less, that value is used as one file name character. The one (or both) parameter(s) that is larger than 8, which must be an integral power n of 2, is represented by the n-th character of the alphabet. For example, if `NT=5` and `NY=1024=`2^{10}, one finds the character `5` followed by the tenth character of the alphabet (`=J`) as a two character block (`--5J--.--`), in all relevant file names. For a list of typical encrypted dimensions and their `NT` $\times$ `NY` equivalence, see the table in section V.D.2.

I.B.1.4 Names of Adjunct Files on the VAX Computer

The other relevant files that reside on the VAX besides the FORTRAN source codes are data files, message files and batch job command files.

INPUT DATA FILE

The input data files can be distinguished by a file name beginning with the character N followed by three more characters and ending with the extension .DAT. E.g.

NR1J.DAT

NPGI.DAT

The second file name character is either R, if this is an input data file for RAM2D1, or P, if this is an input data file for PRAM1. The third and fourth character are the two character block that contains the values of the code parameters `NT` and `NY` encrypted as described in subsection I.B.1.3 .

GRAPHICS DATA FILES

There can be two types of output data files on the VAX. One is the so-called META-file by the name

PLT2.DAT

which is generated by the DISSPLA-graphics subroutines in PRAM1. The other is the data file that the DISSPLA-postprocessing software on the VAX generates with the name

INTSCRT.TMP

when the graphs in PLT2.DAT are requested as laser printer hardcopies. It is the duty of the user to find a means of distinction for these equally named output data files from a series of simulations. It is suggested to rename these files mnemonically. This is easily done by the VAX command RENAME,

type: `RENAME PLT2.DAT` 'new file name'

following the VAX-prompt $.

MESSAGES FILES

Two types of message files can be found in the VAX user directory. Except for a varying file extensions, these files have the same name as the batch job command file (see next paragraph) from which they originated. There are MSG-files. One such file is created if a code generates output due to formatted and/or unformatted write statements. These statements constitute the sole content of this file. The file is identified by its .MSG file extension. For example,

CR1J.MSG

XPGI.MSG

Secondly, there are CPR-files one of which is generated by the CRAY computer every time a job is run. For example,

CR1J.CPR
XPGI.CPR

These files document the batch job execution by recording information such as: program listing, error messages from the CRAY operating system and the CRAY compiler timing information regarding batch job execution, space and cost information and more esoteric information relating to the CRAY computer usage.

JOB FILES

The batch job command files have a name similar to the input data files. The only two differences being the .JOB file extension instead of .DAT, and the initial letter being C or X instead of N. For example,

CR1J.JOB
XR1J.JOB
CP1J.JOB
XP1J.JOB

A first letter X indicates a job file that executes the code associated with it (see second letter of job file name: R for RAM2D1, P for PRAM1). A first letter C indicates a job file that will first compile and assemble the source code before running the newly created executable file.

All file names mentioned above apply to the VAX directory of files [HILFER.FOR]. When a batch job fetches a file (source code or input data file) from VAX storage and transfers it to the CRAY during job execution, the VAX name (specified by `TEXT='---'` on the FETCH command line in the job file) is changed to a CRAY dataset name (as given by `DN='---'` on the same `FETCH` command line).

I.B.1.5 CRAY Dataset Names

SOURCE CODE

The source codes have a three character dataset name when used on the CRAY computer. The first character is R (or P) for RAM2D1 (or PRAM1). The second and third character give the `NT` and `NY` parameter values as described in subsection I.B.1.3. For example,

R1J is RAM2D1 on the CRAY with. $NT=1$, $NY=2^{10}$
PGI is PRAM1 on the CRAY with. $NT=2^7$, $NY=2^9$

Either dataset appears on the CRAY following a FETCH command line in a C—.JOB file and disappears automatically following completion of the job.

EXECUTABLE DATASET

The executable dataset resulting from compilation of either source code is usually kept (SAVE command line in JOB-file) under the same dataset name as its parental source code, but amended by a preceeding X. For example,

XR1J

XPGI

INPUT DATASET

The input dataset to RAM2D1 following a FETCH form the VAX is named

NRAM,

the input dataset to PRAM1 is named

NPRAM1

on the CRAY computer.

OUTPUT DATASET

The output resulting from execution of RAM2D1 is contained in a single CRAY dataset when running the code one-dimensionally. When operating the code two-dimensionally, the number of output datasets is proportional to the number of

z-locations at which field data are kept. All of these data files are saved automatically in the CRAY off-line storage facility.

All output dataset names begin with the letter F followed by eight alphanumeric characters if the file results from one-dimensional code operation, and followed by eleven alphanumeric characters if the file results from two-dimensional code operation. The second and third character in these dataset names are the two character block that contains the values of the code parameters NT and NY encrypted as described in subsection I.B.1.3. The following six characters contain the date at which the execution of RAM2D1 began. In two-dimensional operation three more numerals (a counter) are appended to this same name which number the individual field datasets consecutively as they are created. For example,

F1J101587	(field dataset with arrays dimensioned NT=1, NY=2^{10}, started on October 15, 1987)

FGI101587000	(field datasets with arrays
FGI101587001	dimensioned `NT`=2^8, `NY`=2^9, started
FGI101587002	on October 15, 1987, at different
FGI101587003	z-values)

This counter is 000 for the dataset that contains the list of setup parameters and initial field data. Its purpose is to enable the user to view output data with the diagnostic code PRAM1 immediately as they become available during an extensive run. Such concurrent diagnosis has to be indicated to PRAM1 by setting its input parameter `DONYET` to 0 (`DONYET` should be 1 during regular post mortem diagnosis).

This counter is 001 for the dataset that contains the setup parameters (like -000 dataset), the field data at `ZVAL=0.0` (like -000 dataset), and the timing information gathered at the end of the run (unlike -000 dataset). This counter is 002 for the dataset that contains the field data at `ZVAL=1*ZKEEP`, 003 at `ZVAL=2*ZKEEP`, 004 at `ZVAL=3*ZKEEP`, etc.

MESSAGE DATASET

User defined messages (mostly conditional error messages) from RAM2D1 (PRAM1) are gathered in dataset ERRM (EPRM) which is transferred to the VAX under the name of the current JOB-file but with the file extension .MSG . The other message dataset from each run, the CPR-file, is created by the operating system and not accessable to the user until after it is transferred to the VAX post mortem of the run.

Section I.B.2: Obtaining the Necessary Files

Six files are required to simulate the Raman interaction numerically. These are the FORTRAN source codes

RAM2D1 and
PRAM1

(see subsection I.B.1.1 for full VAX/VMS file names), their respective input data files

NR--.DAT and
NP--.DAT,

and their respective batch job command files

CR--.JOB and
CP--.JOB.

The dashes -- stand for the particular 2-character block as the choice of dimensions, described in subsection I.B.1.3, necessitates.

Unless the user has immediate access (password) to the [HILFER.FOR]- subdirectory it will be necessary to copy these files from there into the user's own directory. The VAX/VMS copy command serves this purpose. When the VAX

```
prompts:        $
type:           COPY DUA107:[HILFER.FOR]RAM2D1C.FOR *.* <
prompts:        $
```

(Should an error message appear, e.g. copy protection violation or insufficient privilege, contact the CCF consultants desk at (202)767-3542 or Godehard Hilfer at (202)767-2028).

```
type:           COPY DUA107:[HILFER.FOR]PRAM1CD.FOR *.* <
prompts:        $
type:           COPY DUA107:[HILFER.FOR]NR--.DAT *.* <
prompts:        $
type:           COPY DUA107:[HILFER.FOR]NP--.DAT *.* <
prompts:        $
type:           COPY DUA107:[HILFER.FOR]CR--.JOB *.* <
prompts:        $
type:           COPY DUA107:[HILFER.FOR]CP--.JOB *.* <
prompts:        $
```

Now all necessary files are in the user's current directory. From this directory the batch job should be submitted in order for the automatic substitution of default values for user disk, default directory etc. in the abbreviated file names as they appear in the batch job command file to work. The message and data files that the job sheds will be send to this directory from which the job was submitted.

Once the dimensionality of the intended simulation is known, the corresponding NT and NY values will have to be encoded as described in subsection I.B.1.3 and filled into all the file names of this section. Remember to insert/replace these two characters also into/in appropriate positions in all file names and dataset names contained in the two JOB-files! Remember also to verify/change all occurrences of NT=--- and NY=--- in both source codes accordingly.

The process of inserting/replacing these characters is called 'editing the file.' The computer software that accomplishes this task is called an 'editor.' A rudimentary description of two selected editors is described below in section I.B.3.

Section I.B.3: Editing Files

I.B.3.1 EDT Screen Editor

In order to make amendments, deletions or any other changes in a file (e.g. an input data file), that file needs to be accessed by an editor program. The preferred editor of the VAX/VMS operating system is called EDT. It accesses any file in the following way. When the VAX

prompts: `$`

type: `SET TERMINAL/VT100 <`

to identify to the editor what industry standard terminal to expect. This setting needs to be made only once after login, not every time the editor is invoked. Giving this setting repeatedly is merely redundant. However, it needs to be set once for the editor to work properly. The terminal used should actually be a DEC VT100 terminal as indicated by the command, or at least emulating such; otherwise, the appropriate setting will have to be found from the VAX/VMS reference manual. Ideally, the user should have a VT240-type terminal to work with. Without its graphics capability it will not be possible to view the output from PRAM1 on the screen. Such terminal is otherwise fully compatible with the VT100 industry standard and will, therefore, work fine in the editor given the above setting. This setting is taken by the VAX without any special response, it just

prompts: `$`

Then

type: `EDIT/EDT` 'filename.extension' <

and fill in for 'filename.extension' the name of the file that shall be edited.

CREATING/EDITING A NEW FILE

The same command

```
EDIT/EDT 'filename.extension'
```

can also be used to create a new file by filling in a filename that is not yet in the directory. (To see which files are already in the directory see below in section I.B.4.)

In that case the system

prompts:
```
Input file does not exist
[EOB]
*
```

The star indicates that the editor is in its default mode which is the line editing mode. However, the power and primary function of EDT is its screen editing capability. To change to screen editing mode

type: `c <`

following the star prompt. Then the screen will be erased and in the top left corner appears the [EOB] indicating the end of the buffer. Buffer is the name for storage space that is volatile. The characters stored in it will disappear after the process to which the buffer belongs is terminated unless the buffer is purposely saved. Anything that the file contains, can now be typed into the buffer. The 'end of the buffer' indicator moves automatically down the screen as characters are inserted. The buffer is saved and becomes the desired file if the editing session is ended with the **END** instruction. The alternative would be to finish editing with the **QUIT** instruction where upon the buffer is discarded leaving no trace of the editing session whatsoever. To finish either way

type: ˆ z (*Ctrl z* ; i.e. while holding the *Ctrl* key on the keyboard down type a 'z', then release both keys; no additional return key stroke is necessary; although it would do no harm)

The editor will return to the line editing mode that

prompts: *

To exit

type: exit < (to exit and to save the buffer content in a disk file)

or

type: quit < (to exit and to lose the buffer content)

EDITING AN EXISTING FILE

If the 'filename.extension' in the EDIT/EDT command line

EDIT/EDT 'filename.extension' <

matches one, or several, entries in the current directory the editor will access the one of these files that has the highest version number. Access is accomplished when the computer

prompts:
```
1 ----- 'text of first line in file'---------
*
```

This star is the line editor mode prompt.

type: c <

to get into screen editor mode.

SCREEN EDITING TOOLS

Most screen editing consists in moving the cursor to the desired position on the screen and then entering characters there, by typing them, or deleting characters there. For this the essential tools are the special keyboard keys:

arrows (*left*, *right*, *up*, *down*; move the cursor one field at the time by pressing the key shortly; scroll the cursor in that direction by holding the key down)

delete (erases a character to the left of the current cursor position)

PF4 (erases a whole line following the current cursor position at once)

PF1 PF4 (undoes the last delete of the *PF4* key)

The set of 18 keys in the lower right corner of the keyboard is called keypad. Its keys, designated in this manual by a preceding *P* (e.g. *P4* is keypad key 4), have special functions in EDT (e.g. *PF1* and *PF4* described above). To view a description of these functions press the *PF2* key. For the extensive user of the VAX, it is desirable to memorize the use of the keypad. For the occasional user it shall suffice to mention the block delete/move procedure: select desired block of text by marking invisibly one end by hitting *P.* (that is the . key on the keypad) (undo erroneous use of that key by pressing *PF1* followed by *P.*); Use the arrow keys to move the cursor to the other end of the intended block boundary; press *P6*; now the block is moved from the displayed text buffer into a hidden text buffer. From there it can be copied to the current cursor position as often as desired by pressing *PF1* followed by *P6*. The block will remain in the hidden buffer until another block delete overwrites it, or until the editor is exited.

Standard editing shows a maximum of 80 characters per column. To view CPR-files it is appropriate to display 132-characters per line. To change to that format

type: *PF1 P7* `SET SCREEN 132` *PEnter* (*PEnter* is the *enter*-key on the keypad)

Very, useful particularly when viewing a CPR-file, are the EDT-commands for fast scroll to end or beginning of the file:

type: *PF1 P4* (for fast scroll to the end of the file),

type: *PF1 P5* (for fast scroll to the beginning of the file),

The key *P8* is not quite that fast, but still faster than the arrow keys, in scrolling forward or backward in the file. If preceded by *P4*, *P8* will scroll 16 lines forward, if preceded by *P5*, *P8* will scroll 16 lines backward. The direction key *P4* or *P5* needs to be pressed only once. *P8* can be applied repeatedly thereafter.

These are the basic EDT screen editing commands that the user will need. Further detail can be found on line (press PF2) or in the VAX/VMS reference manual on EDT.

I.B.3.2 TEDI Line Editor

The widely used line editor TEDI shall be introduced because of its convenient pattern search and replace operation. Line editing consists in displaying and modifying a particular line or several lines at the same time.

For the TEDI editor to access the file 'filename.extension',

type: `TEDI` 'filename.extension' <.

This

prompts: `DUA107:[DIRECTORY]`filename.extension`;1 ---LINES`

*

The star is, just like in the EDT editor, the line mode prompt.

TEDI commands consist of one or a few acronymic letters accompanied by one to three line numbers separated by commas and, separated by semicolons, followed by one or two character strings, depending on the particular command.

The TEDI editor can list and replace efficiently all occurrances of a given character pattern. This is useful when checking and/or changing the dimensionality of the field arrays in the source codes. To accomplish this

type: `TP1,500;NY=; <`

following the star prompt. This instructs the computer to type all lines between line 1 and line 500 in the currently accessed file that contain the pattern: `NY=`. Note that TEDI distinguishes letters also by their capitalization. To search the whole file one needs to replace 500 by a number equal to or larger than the total number of lines in the file or, if unknown, to replace 1,500 by the wildcard symbol *. For example, 'tp*;NT='. The command accronyms can be small or large case letters. The last semicolon may be and was omitted.

To replace all occurrences of NY=1 by NY=512, for example,
type: `RP1,500;NY=1;NY=512; <`

The type pattern (TP) command preceeding the replacement (RP) is somewhat tedious but efficient if there is any doubt about possibly unwanted replacements like: ISNY=1. Therefore, TP should be employed to make sure that the intended pattern string is unique.

Portions of a file can be viewed by the type command:

```
T1,500 <
```

would scroll lines 1 through 500 across the screen. The command

```
T* <
```

scrolls the whole file. An individual line (e.g. line 500) can be deleted by

```
DL500 <
```

Several lines are deleted by giving the range (e.g. line 1 through 500)

```
DL1,500 <
```

Caution! Deletes cannot be restored in TEDI except for the price of giving up all the other editing that was done beforehand through an emergency exit from the editing session (type: quit).

New lines can be added before (BL) or after (AL) any specified line number. For example,

```
BL1 <
```

starts the insertion of lines before the current line number 1. Insertion mode is indicated by the '>'-prompt. All following characters will be inserted sequentially as typed. Another new line is inserted with every return '<'. Insertion mode is ended by typing a '.' by itself on a new line.

A detailed description of the TEDI editor is on file in the CCF consultants office or can be purchased from the CCF operator desk.

Section I.B.4: Directories / Delete / Purge

I.B.4.1 VAX

DIRECTORY

A listing of the directory of files on the VAX can be viewed in the following way: Change, if necessary, the directory information that is contained in the omitted portion of the complete file name to the desired directory `DISK:[USER.SUBDIRECTORY]`. To this end

type:	`SET DEFAULT DISK:[USER.SUBDIRECTORY] <`
prompts:	`$`

Then the listing of files in that subdirectory appears after you

type:	`DIRECTORY <`

may be shortened to `DIR <`.

The `DISK:` specification may be omitted if unchanged. The `.SUBDIRECTORY` specification has to be omitted to see the main `[USER]` directory list of files. Multiple level subdirectories can be listed in the same way by continuing the path of directories starting with the main directory in the analog fashion:

```
SET DEFAULT DISK:[USER.SUBDIR.SUBSUBDIR.SUBSUBSUBDIR] <
```

The plain listing of all files can be more elaborate by means of file name portion, filters, and options following the `DIRECTORY` command. For details

type:	`HELP DIRECTORY <`

which can be terminated by one or several '<' returns.

DELETE

To delete an entry from the directory of files and thereby destroy that file

type:	`DELETE 'filename.extension;version' <`

The specified file name is removed from the default directory (see I.B.1.1) only. It is necessary to specify the version number otherwise no deletion will take place but rather an error message will appear on the screen. The three pieces in the name of the file: filename, extension, and version can be substituted with the wild card character '*' in order to generalize the command to delete all files that match the specification except for the name piece represented by the '*'. For example,

type:	`DELETE NRAM.DAT;*`

to delete all versions of the file NRAM.DAT (contrary to `PURGE NRAM.DAT` which leaves the highest version). For example,

type: `DELETE N*.DAT;*`

to delete all files whose names begin with the letter N, by the file extension .DAT from the directory. For more sophisticated usage of the DELETE command

type: HELP DELETE

which can be exited by one or several '<' returns.

PURGE

To purge the default directory of files is to remove all file versions except for the last (highest) one. To purge the current VAX default directory simply

type: PURGE <

The PURGE command can be made more specific. For details

type: HELP PURGE <

which can be exited by one or several '<' returns.

I.B.4.2 CRAY

The simple functions of listing the file directory, purging it and deleting particular entries are somewhat time consuming on the CRAY computer due to the batch job setup. Therefore, a batch job has to be submitted to accomplish these tasks. How to submit a batch job will be demonstrated in the next Section I.B.5: Running a Batch Job.

DIRECTORY

The listing of the files in the user directory on the CRAY disk is obtained in the CPR-output file of any CRAY job if the job command file contains the command line with the command:

```
AUDIT..
```

This is usually the case with every batch job, hence, the need for at-will CRAY directory information is small. Nevertheless, the job command file

DUA107:[HILFER.FOR]CAUDIT.JOB

can be copied to do only that when submitted as a batch job.

DELETE

In order to delete a file in the CRAY directory a batch job command file needs to be submitted that contains the appropriate DELETE command line. For example,

```
DELETE,PDN='filename'. .
```

The user may wish read the details of DELETE command line in the CRAY operating system (COS) manual. The quickest path for the new user is simply to copy the file DUA107:[HILFER.FOR]CDELET.JOB into the current directory, to change the file name contained in it as desired, and to submit it for execution. Notice that the '-' character serves as the wild card character of COS representing any string of characters.

PURGE

In order to purge files in the CRAY directory, i.e. delete all versions but the latest of each file, a batch job command file needs to be submitted to the CRAY that accomplishes to delete in a selective way. For example,

DELETE,PDN=-,ED=-1. .

The user can find the details of the DELETE command in the COS-manual. A simpler path is to copy the file DUA107:[HILFER.FOR]CPURGE.JOB into the user directory, and to edit the contained file names such that all file names to be purged are covered by the specified file name pieces in combination with wild cards. Recall that on the CRAY the symbol '-' is the wild card for any string of characters.

Section I.B.5: Running a Batch Job

I.B.5.1 The Batch Job Command File

The execution of a computation on the CRAY computer as a batch job requires several steps which are listed as command lines in the batch job's .JOB-file . Once this file is transferred to the CRAY it will be queued in the batch job queue. When its turn for execution comes around, the operating system will execute all command lines sequentially, waiting for each command to finish before picking up the next one. Should a terminal error occur, execution will be stopped. A the end of each job a log-file will be sent to the user's VAX directory.

There are a few rules concerning the form of the batch job command file: Beginning in column 1, every line must start with a command verb that is known and accepted by the operating system. Every line must end with a period ('.'). Several parameters may follow the command verb separated by commas. The first command line in the file must be the JOB-statement:

```
JOB,JN='job name'.
```

(CBATCH processing waives this requirement, see section V.B.4. The name that the job shall have has to be inserted. The second command line must be the ACCOUNT-statement:

ACCOUNT,AC='account number',US='user number',UPW='user password'.

(CBATCH processing waives this requirement, see section V.B.4 which has to be completed by the three appropriate fill-ins: account number, user number, and user password. The next command lines contain the desired CRAY action followed by the command line:

```
EXIT.
```

Note that all JOB-files that are copied from the DUA107[HILFER.FOR] directory lack the JOB and ACCOUNT command line which will have to be supplied by the user.

I.B.5.2 Submitting a Batch Job

It is recommended to preceed the submission of the first batch job, when the VAX prompts: $

with the following VAX command,
type: CRAY SET TERMINAL INFORM <

This will inform the CRAY computer of the location of the user's terminal and, hence, enable forwarding of the messages that accompany the execution of the job.

For the actual submission of the JOB-file
type: CRAY SUBMIT 'filename'.JOB <

where the JOB-file's filename has to be inserted. This will queue the job file for transfer to the CRAY and subsequently queue it for execution. For example

```
type:      CRAY SUBMIT CAUDIT.JOB <
prompts:   $
           % CX-S-SUB_OK, Job:  CAUDIT queued for submission
           $
           VAX TO CRAY: % SYSTEM-S-NOMRAL, normal successful completion
           VAX TO CRAY: FILE=CAUDIT
VAX TO CRAY: 4608 BYTES TRANSFERRED
$
```

which are the standard messages of verification for the queuing for submission and for the transfer of the JOB-file for the CRAY computer.

I.B.5.3 Batch Job Execution and Termination

The execution of the job is determined by the CRAY operating system. During the execution of RAM2D1 and PRAM1 other files are transferred from the VAX to the CRAY. Each transfer is accompanied by a message of the type

```
VAX TO CRAY: % SYSTEM-S-NOMRAL, normal successful completion

VAX TO CRAY: FILE=NRJ1

VAX TO CRAY: 4608 BYTES TRANSFERRED
```

Progress of execution can be monitored through on-demand status messages. For this purpose
type: `CRAY STATUS/OWN <`
prompts:

```
cray  system status  EIORS      PRIMARY   17-feb-1988  11:39:50.19
jsd   dc  dataset  class   status   pri   used limit   length  id    tid
12596 IN  CAUDIT   SMALL   QUEUED   6.0      0    60      512  V2  HILFER
```

the explanation of each detail for which all would break the frame of this manual. The important points, however, are the STATUS, the number of seconds USED, and the number of seconds LIMIT for the job. Those three items are self-evident.

The termination of a job occurs usually automatically when the EXIT. command line in the JOB-file is executed. Such normal (and other unusual) termination is indicated by the transfer of the CPR-file from the CRAY to the VAX as notified of by a message of the following type:

```
CRAY TO VAX: % RMS-S-NORMAL, normal successful completion

CRAY TO VAX: FILE=1DUA107:[HILFER.FOR]CAUDIT.CPR;1

CRAY TO VAX: 1706 BYTES TRANSFERRED
```

Another definite indication is when the response to the status request explained just above is responded by only the first two headlines, showing no job sequence number. The successful transfer of the CPR-file does not indicate that the program ran successfully. This can only be seen from the bottom portion of the CPR-file.

Unusual termination can be due to, e.g., programming errors, command line errors, too small a time limit (job needs more CPU-time than the allocated amount; =60sec by default), forced by the user and other reasons. When a submitted job needs to be stopped, obtain at first the jsq-number from the `CRAY STATUS/OWN` report, then type: `CRAY KILL'jsq-number' <`.

This will result in the termination of the job that is documented as such in the subsequently issued CPR-file.

I.3.5.4 VAX Job Interruption

An emergency stop of any VAX DCL-command can be forced by typing `^ Y` (=*Ctrl* `Y`). This causes the VAX computer to interrupt whatever it was engaged in and to return to the $-prompt, ready for a new command.

CHAPTER II

RUNNING RAM2D1 AND PRAM1

To perform the actual Raman amplifier simulation, one only needs to submit a JOB-file that compiles the source code RAM2D1 and runs the resulting executable file. Hence

type: `CRAY SUBMIT CR--.JOB`

where the '--' holds the place for the appropriate dimensionality characters (see PART I.B).

To diagnose the results of a Raman amplifier simulation, one only needs to submit a JOB-file that compiles the source code PRAM1 and runs the resulting executable file. Hence

type: `CRAY SUBMIT CP--.JOB`

where the '--' holds the place for the appropriate dimensionality characters (see PART I.B).

As a reminder, we repeat several points: 1) ensure that RAM2D1 and PRAM1 have the desired dimensions in all its subroutines; 2) ensure that the input data file NR--.DAT, contains the desired input parameters; 3) ensure that the JOB-file transfers the desired set of files.

If all appears well, submit the job as shown above. Monitor the job progress by reading the messages on the screen and/or inquire the status as described in section I.B.5. Job termination is indicated by the transfer of the CR--.CPR file from the CRAY to the VAX. Use EDT's 132 column screen editing mode to check the CPR-file for error-free execution of the whole job. In case of error messages, turn to PART V.C. or call Godehard Hilfer at (202)-767-2028.

In a series of simulations, it is unnecessary to recompile the source code for each simulation over again. Instead one can copy, or create, the XR--.JOB file and type:

```
CRAY SUBMIT XR--.JOB
```

to submit the next simulation. The XR--.JOB file is a copy of the CR--.JOB that lacks the compilation and loading command lines. Hence, it will only run the executable dataset XR--. The corresponding CPR-file is XR--.CPR .

CHAPTER III

VIEWING THE RESULTS

PART III.A TERMINAL OUTPUT

The data file that arrives in the VAX user directory at the end of PRAM1's execution, PLT2.DAT, is a device independent graphics data file generated by the DISSPLA library routines contained in PRAM1. In order to see the graphs on the terminal screen, DISSPLA postprocessing software needs to be applied. For this purpose, unless previously done during this login,

```
type:       GRAPHICS_LOGICALS
prompts:    $
type:       PUBLIC_LOGICALS
prompts:    $
```

(to make use of site specific software and setups)

Then attach the data file to the post-processing software and run it

```
type:       RUN VT240$POP <
prompts:    THIS IS THE VT240 POST-PROCESSOR ENTER YOUR POST-
            PROCESSOR DIRECTIVES OR A CARRIAGE-RETURN FOR
            DEFAULTS
```

To view all graphs one only needs to

```
type:       <
```

a carriage return. This will produce the first graph on the screen. Another carriage return will erase the first graph and draw the second graph. Any more carriage returns will sequentially display the rest of the graphs until the last carriage return

prompts:
```
END OF DISSPOP 2.2 -- 2057 VECTORS IN 1 PLOTS RUN ON 2/17/88
USING SERIAL NUMBER 60 AT NRL PCC VAX PROPRIETARY
SOFTWARE PRODUCT OF ISSCO, SAN DIEGO, CA
$
```

which automatically finishes the post-processing.

To be more selective in which graphs shall actually be displayed, one has to enter those graph numbers explicitly when asked for the post-processor directives. For example,

type: `DRAW=5-9,12,17-20 <<`

to display graphs numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20. Notice that it takes two carriage returns to continue the postprocessing. If a few in a large series of graphs shall be excluded from viewing, one can, rather than listing all the others, 'delete' those particular graphs from the display. Hence,

type: `DELE=1-4,10,11,13-16,21-END <<`

to display the same graphs numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20 as before. Note, deletes supersede draws, and the sequence of listing is immaterial.

For more details, see the DISSPLA users manual part F. DISSPOP post-processing.

PART III.B HARDCOPIES

Section III.B.1: Printed Graphs

The data file that arrives in the VAX user directory at the end of PRAM1's execution, PLT2.DAT, is a device independent graphics data file generated by the DISSPLA library routines contained in PRAM1. In order to obtain the graphs on paper, DISSPLA postprocessing software needs to be applied. For this purpose, unless previously done during this login,

type: `GRAPHICS_LOGICALS`

prompts: `$`

type: `PUBLIC_LOGICALS`

prompts: `$`

(to make use of site specific software and setups)

Then attach the data file to the post-processing software and run it.

type: `RUN LN01$POP <`

prompts:
```
THIS IS THE VT240 POST-PROCESSOR ENTER YOUR POST-
PROCESSOR DIRECTIVES OR A CARRIAGE-RETURN FOR
DEFAULTS
```

To process all graphs for printing one only needs to

type: <

a carriage return. This will produce a new data file called INTSCRT.TMP in the user's directory which then can be printed straightforwardly. At the end of processing the computer

prompts:
```
END OF DISSPOP 2.2 -- 2057 VECTORS IN 1 PLOTS RUN ON 2/17/88
USING SERIAL NUMBER 60 AT NRL PCC VAX PROPRIETARY
SOFTWARE PRODUCT OF ISSCO, SAN DIEGO, CA
$
```

which automatically finishes the post-processing.

To be more selective in which graphs shall actually be post-processed, one has to enter those graph numbers explicitly when asked for the post-processor directives. For example,

type: `DRAW=5-9,12,17-20 <<`

to graph frames numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20. Notice that it takes *two* carriage returns to continue the postprocessing. One can also delete particular graphs,

type: `DELE=1-4,10,11,13-16,21-END <<`

to process the same graphs, numbered 5, 6, 7, 8, 9, 12, 17, 18, 19, 20, as before. The command `DELEte` supersedes `DRAW` and the sequence is immaterial. For more details, see the DISSPLA users manual part F. DISSPOP post-processing.

The device specific file INTSCRT2.TMP (last version is default version number) can be send directly to the CCF or A49 laser printer. Thence,

type: `LASER/PLOT/CCF/NOTIFY INTSCRT2.TMP <`

or

type: `LASER/PLOT/A49/NOTIFY INTSCRT2.TMP <`

The terminal will notify of the completion of printing with a beep and a message. The print-out can then be picked up in building A49 either at the CCF-desk (output from the CCF laser printer) or in the Remote-Print-Station room in building A49 (output from the A49 laser printer).

Section III.B.2: Printed ASCII-Files

The printing of regular text files is done with either one of the following two commands,

type: `LASER/PORT/CCF/NOTIFY <`

for a print-out on the CCF laser printer in building A49 or

type: `LASER/PORT/CCF/NOTIFY <`

for a print-out on the 'remote print station' laser printer in building A49. Both commands will cause the terminal to notify of the completion of the printing job with a beep and a message. For files with lines of more than 80 characters length, the printing can be turned by 90 degrees from the high format to the wide format of the $8.5 \times 11 inch$ pages. For this

type: `LASER/LAND/CCF/NOTIFY <`

or

type: `LASER/LAND/A49/NOTIFY <`

CHAPTER IV

SUSTAINING OPERATIONS

PART IV.A FILE STORAGE

Section IV.A.1: VAX Disk

The most essential and only necessary storage device for the code operation is the VAX disk storage space. Since it is the default storage location, no special steps need to be taken to store files in the VAX computer on those disks. The essential files (source codes, input data files, batch job command files, and graphics output data files) are stored here. However, the space allocation for the user is limited and can be restrictive when producing graphs. To obtain a quotation of the allocated and used disk storage space,

type: `SHOW QUOTA` <

which will respond with a message that gives the total allocated storage space, the portion used, the portion remaining, and the overflow margin. If the allocated space is continuously insufficient, turn to the CCF system manager or consultant for an increase. If the shortage of storage space is expected to be only occasional, then turn also to the consultant for access to the so-called scratch disk. This whole disk is available on a first come, first serve basis. Files will be kept on it for at least 24 hours but at most 48 hours. Hence all post-processing and printing of particularly large graphics data files can be done before the system software wipes out all scratch disk files routinely.

The size of individual files can be obtained when listing the directory of files by specifying `/SIZE=USED` in the `DIRECTORY` command. Hence,

type: `DIR/SIZE=USED` 'filename.ext' <

Furthermore, the date the file was created can be inquired by specifying /DATE=CREATED in the `DIRECTORY` command. For this

type: `DIR/DATE=USED` 'filename.ext' <

This way, a more selective clean up of the user directory is possible, hopefully, maintaining sufficient space for all user activity.

VAX storage space is measured in units called BLOCKS,

1 Block = 512 Bytes.

(Recall 1 Byte = 8 Bits)

The price for VAX disk storage is currently $0.00016 per Block per day.

Section IV.A.2: VAX Tape

For more economical storage and to keep the VAX disk quota sufficient it is recommended to store files on VAX tape. This is called archiving the file. For on-line documentation regarding the archiving options

type: `HELP ARCHIVE` <

The most important features will be listed here.

To archive a file means that that file is physically removed from the VAX disk to the tape. Hence, to keep a copy on the disk an explicit copy must be made,

type: `COPY` 'file-to-archive.ext' 'remaining-file.ext' <

To archive the desired file

type: `ARCHIVE` 'file-to-archive.ext' <

This removes the file from the directory.

To list the files that had previously been archived

type: `ARCHIVE/DIR` <

Since the archiving is done overnight by the operating system, the newly archived file, although gone from the directory, does not yet show up as archived. It is queued for archival. To list the files awaiting archival

type: `ARCHIVE/LIST` <

If an error occurred, the file can be retrieved from this queue of files bound for archival; For this

type: `ARCHIVE/CANCEL` 'file-to-archive.ext' <

To remove a file from the archive

type: `REMOVE` 'archived-file.ext' <

prompts: `REMOVE DUA107:[USER.DIR]`'archived-file.ext;x' ?

type: **yes** <

then the file disappears.

VAX storage space is measured in units called *blocks*,

1 *block* = 512 *bytes*.

(Recall 1 *byte* = 8 *bits*)

The price for VAX disk storage is currently $0.00001 per block per day.

Section IV.A.3: CRAY Disk

As was mentioned in the introduction, all files on the CRAY computer are volatile. That is, they will not be stored by default, rather they will be destroyed by default. Therefore, it is necessary to save explicitly all files that need to be saved. To this end, after compilation of the source codes, the executable files are saved by the batch job command file on CRAY disk. Unless otherwise specified the system's storage (save) commands save the datsets on the CRAY disks. Examples for two procedures are:

```
SAVE,DN=XR--.
```

(example for batch job command line for storage),

```
CALL SAVE(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D)
```

(example for FORTRAN statement for file storage, where `'DN'L,DTFL1D` indicates that the character variable `DTFL1D` is the dataset name when the program is running, and `'PDN'L,DTFL1D` is the permanent dataset name by which the file will be listed on the disk.

Storage space on the CRAY is not allocated individually, but always on a first come, first serve basis. Operating system software ensures, by moving big, old files automatically from disk to tape, that there is always storage space available. When listing the directory file names by means of the AUDIT. command, the right hand column indicates whether the listed file is on disk (on-line) or on tape (off-line).

The price for CRAY file storage is the same as that for VAX file storage. Hence, CRAY disk storage is charged at $0.00016 per *block* per day.

(Recall 1 *block* = 512 *bytes*.
1 *byte* = 8 *bits*)

The standard measure for CRAY storage is 1 *sector* = 8 Blocks = 512 CRAY words of 64 *bits* each)

Section IV.A.4: CRAY Tape

As was mentioned in the introduction and in section IV.A.3 above, all files on the CRAY computer are volatile. They will not be stored by default; they will be destroyed by default. Therefore, it is necessary to explicitly save all files that need to be saved. To this end, the programs RAM2D1 (and PRAM1) contain CRAY operating system calls that save the data files on CRAY tape (called off-line). To save a file on CRAY tape, one has specify to that location on the SAVE command line. For example:

```
      CALL SAVE(IRRE,'DN'L,DTFL1D,'PDN'L,PDN1D,'RESIDE'L,'OFFLINE'L)
```

This is a CRAY FORTRAN statement for file storage, where `'DN'L,DTFL1D` indicates that the character variable `DTFL1D` holds the dataset name when the program is running, and `'PDN'L,DTFL1D` contains the permanent dataset name by which the file will be listed on the disk. Residency off-line is explicitly mentioned.

The storage space on the CRAY tape is not allocated individually, but always sequentially used on first come first serve basis available. Operating system software ensures, by moving big old files automatically from disk to tape, that there is always storage space available. When listing the directory file names by means of the `AUDIT.` command, the right hand column indicates residency of the file on CRAY disk as on-line.

The price for CRAY file storage is the same as that for VAX file storage. Hence CRAY disk storage is charged at $0.00016 per *block* per day.

(Recall 1 *block* = 512 *bytes.*
1 *byte* = 8 *bits*)

The standard measure for CRAY storage is 1 *sector* = 8 *blocks*)

PART IV.B OPERATOR RELIEF

Section IV.B.1: Login Command File

Many settings and definitions should be repeated every time a user logs into the front-end VAX computers. To save the user the typing effort of these settings, it is possible and recommendable to let the computer repeat this sequence of definitions and commands automatically. This can be done by means of the LOGIN.COM file. This file, which has to be in the user's root (login default) directory, is a command file that the computer executes automatically every time the user's logs into the VAX computer. For details on the meaning and syntax of command lines in this file, see the VAX/VMS DCL-manual. The following list is an example for some of the login commands and definitions which typically appear in a LOGIN.COM-file.

COMMANDS

```
$ SET TERMINAL/VT100
```

informs the operating system of the terminal's industry standard

```
$ GRAPHICS_LOGICALS
```
invokes site-specific system definitions
```
$ PUBLIC_LOGICALS
```
invokes site-specific system definitions
```
$ CRAY SET TERMINAL INFORM
```
advise operating system to output CRAY messages to terminal
```
$ SHOW TIME
```
show current time on terminal
```
$ SHOW QUOTA
```
show current VAX disk storage distribution of the owner.

DEFINITIONS

a) of acronyms of customized directory lists of file groups

```
$ DIRALL :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING
  $ DCPR :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.CPR
  $ DDAT :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.DAT
  $ DFOR :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.FOR
  $ DJOB :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.JOB
   $ ETA :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *PLT*.DAT
  $ DMSG :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.MSG
  $ DTMP :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *.TMP
```

b) of acronyms of customized directory lists of files of a standard dimensionality

```
   $ DEE :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *EE.*
   $ DG1 :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *G1.*
   $ DH1 :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *H1.*
   $ DI1 :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *I1.*
   $ DJ1 :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *J1.*
   $ DK1 :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *K1.*
   $ D1G :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1G.*
   $ D1H :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1H.*
   $ D1I :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1I.*
```

```
    $ D1J :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1J.*
    $ D1K :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1K.*
    $ D1L :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1L.*
    $ D1M :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1M.*
    $ D1N :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1N.*
    $ D1O :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1O.*
    $ D1P :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *1P.*
    $ DGI :== DIRECTORY/SIZE=USED/DATE=CREATED/TRAILING *GI.*
```

c) of acronyms of other customized commands that have been explained elsewhere in this manual

```
        $ H :== SET DEFAULT DUA107:[HILFER]
     $ HFOR :== SET DEFAULT DUA107:[HILFER.FOR]
  $ PLOTMPL :== LASER/PLOT/CCF/NOTIFY INTSCRT2.TMP
  $ PLOTMPS :== LASER/PLOT/A49/NOTIFY INTSCRT2.TMP
   $ POP240 :== RUN VT240$POP
     $ POPL :== RUN LN01$POP
```

Section IV.B.2: Edit-Aid

Certain customized features of the EDT editor can be made standard if the LOGIN.COM-file (see section IV.B.1) contains the following definition.

```
        $ E :== EDIT/EDT/COMMAND=DUA107:[USER]EDTINI.EDT
```

This shortens the command line that starts the EDT editor to the letter *e* plus the file-name and at the same time implements the definitions contained in the file [USER]:EDTINI.EDT for which an example follows:

```
DEFINE KEY GOLD N AS ''EXT SET SCREEN 80.''
```

(defines the two key strokes PF1 N to change the terminal display to 80 column width)

```
DEFINE KEY GOLD W AS ''EXT SET SCREEN 132.''
```

(defines the two key strokes *PF1 W* to change the terminal display to 132 column width)

```
SET SCREEN 72
```

(sets the display width to 72 columns)

```
SET WRAP 72
```

(sets the editor's feature of swapping terminal entries beyond column 72 into the next line)

```
SET MODE CHANGE
```

(causes the EDT editor to change to screen editing mode automatically at the onset of the editing session)

Section IV.B.3: VAX/CRAY Status

It is useful to define mnemonic acronyms for the monitoring functions of the VAX and CRAY computer systems. Some frequently used definitions from the LOGIN.COM-file of the author are:

CRAY:

```
$ CSO :== 'CRAY STATUS/OWN
```

(see subsection I.B.5.3 for details)

VAX:

```
$ CNRL == ''MON CLU/INT=1''
```

(to monitor the work load distribution between the four front-end VAXes)

```
$ MSYS == ''MONITOR SYSTEM"
```

(to monitor the CPU (computing), memory, and I/O (data input/ouput) load of the VAX in use)

```
$ MTOP == ''MONITOR PROCESSES /TOPCPU"
```

(to monitor the list of VAX-processes with the highest CPU demand)

```
$ SQ == ''SHOW QUEUE/DEVICE"
```

(to list the queue entries on all devices)

Section IV.B.4: CRAY Grease

There are ways of making the interaction with the CRAY computer more convenient and speedy. For example, the submission of batch jobs can be simplified by using the command `CBATCH` in combination of several mnemonic acronyms. `CBATCH` is a CCF command that allows one to eliminate the obligatory `JOB` and `ACCOUNT` command line from all job files by automatically attaching a file that contains those two command lines in an encrypted form. The file is named $CRAY$.ACCOUNT and resides in the user's root directory. It is automatically generated the first time `CBATCH` is used. For details on this command type: `HELP CBATCH <`

Every batch job that is submitted to the CRAY computer is placed into a class of jobs that are similar in memory size and priority requirements. There are four classes with regard
to job-size: small = up to 511000 CRAY words
medium = up to 1023000 CRAY words
large = up to 1535000 CRAY words
x-large = up to 3071000 CRAY words

and three subclasses of the four with regards to the requested
priority level: express charged 1.5,
normal charged 1.0,
defered charged 0.7

times the price of $900.00 per hour of CPU usage.

To submit a job quickly and conveniently while at the same time specifying these details, the following definitions may be found helpful when present in the LOGIN.COM-file:

```
$ CBDS :== CBATCH/JUS=DEFER/MFL=511000/AC=-----
$ CBDM :== CBATCH/JUS=DEFER/MFL=1023000/AC=----
$ CBDL :== CBATCH/JUS=DEFER/MFL=1535000/AC=----
$ CBDXL :== CBATCH/JUS=DEFER/MFL=3071000/AC=----
$ CBNS :== CBATCH/JUS=NORMAL/MFL=511000/AC=----
$ CBNM :== CBATCH/JUS=NORMAL/MFL=1023000/AC=---
$ CBNL :== CBATCH/JUS=NORMAL/MFL=1535000/AC=---
$ CBNXL :== CBATCH/JUS=NORMAL/MFL=3071000/AC=---
$ CBXS :== CBATCH/JUS=EXPRESS/MFL=511000/AC=---
$ CBXM :== BATCH/JUS=EXPRESS/MFL=1023000/AC=---
$ CBXL :== CBATCH/JUS=EXPRESS/MFL=1535000/AC=--
$ CBXXL :== CBATCH/JUS=EXPRESS/MFL=3071000/AC=--
```

which have to be completed with the appropriate charge account number following the AC= parameter. With these acronyms, the submission of a batch job becomes quite simple. Since the CBATCH command expects a .JOB-file, the file extension (which is .JOB) can even be omitted. So, for example, to submit a small batch job that deletes a file from CRAY storage, one need only type: `CBDS CDELET <`

or to run a particular full scale Raman interaction simulation

type: `CBDXL CRGI <`

These batch jobs are assumed to take less than 60 seconds of CPU time for completion. Should more CPU time be required, provide the /T=400 parameter to allow maximally 400 seconds of CPU time for execution. For example

```
CBDXL/T=400   CRGI
```

Note: be generous with the time limit to aviod having to rerun (and pay again) the whole job for lack of time allocation.

If the maximal memory requirement is known from the CPR-file of a previous run with the same dimensions, the right job class can be chosen. To choose the right class, one should consider also the system's limit of how many jobs of a certain class can run simultaneously. These are:

service class	resource class	max. jobs	priority
express	small	12	9
express	medium	6	9
express	large	2	9
express	xlarge	2	9
normal	small	10	6
normal	medium	4	6
normal	large	2	6
normal	xlarge	2	6
normal, long time	small	10	6
normal, long time	medium	4	6
normal, long time	large	2	6
normal, long time	xlarge	2	6
defered	small	5	3
defered	medium	2	3
defered	large	2	3
defered	xlarge	2	3

The meaning of the `normal, long time` class is subtle and should not concern the user, except that the job will be counted in the long time class if more than 300 seconds CPU time are requested. To find out how full the desired class currently is

type: `CRAY <`

following the standard VAX/VMS DCL prompt: $. Then the screen

prompts: `CRAY>`

type: `STATCLASS <`

which will be responded with a table of the current job class demand. To scroll down

type: `+ <`

To scroll up

type: `- <`

To exit the display

type: `EXIT <`

prompts: `$`

Section IV.B.5: Money Savers

Some methods follow that will reduce the cost of computing. Generally, The most important money saver is the algorithm itself. To use the most efficient numerical scheme for obtaining the results of any computation is the key to low cost. Other methods usually provide only a fraction of the possible savings. Some of those methods are mentioned here.

The program RAM2D1 comes in two versions: RAM2D1C and RAM2D1D. They differ only in their memory requirements. In two-dimensional operation the three megaword random access memory (RAM) capacity of the CRAY X-MP machine often is exceeded. For simulations of this size one needs to use RAM2D1D. That version keeps only two work arrays in memory and stores intermediate results of the computation on CRAY disk memory at the expense of voluminous data input and output. The associated high I/O charges can be saved by using RAM2D1C whenever possible.

One can save 30 percent of the CPU-charges by running the job with priority=deferred rather than priority=normal (see section IV.B.4 for details). This change did not appear to alter the job turnaround time noticeably. Running a small job with priority=express is more expensive but also not very noticeable in terms of job turnaround time, since only the CPU processing is prioritized, not the file transfer.

Savings result also from the use of tape storage rather than disk, for long term file storage. These savings can be significant if the file is stored for a long length of time. The biggest savings are obtained if the file under consideration for storage can be discarded altogether rather than stored. This decision requires extreme prudence. Here, one can easily save pennies but waste dollars when having to regenerate the discarded dataset.

PART IV.C TROUBLE SHOOTING

For all sorts of invincible obstacles, the user will find ample support from the CCF consultants. They can be reached by phone: (202) 767-3542 or (202) 767-1374. One can also send them a message over the VAX MAIL facility (type: HELP MAIL for details; when prompted for the recipient type: CONSULTANT). For all problems, especially regarding the codes RAM2D1 and PRAM1, the user may wish to call Dr. Godehard Hilfer at (202)-767-2028.

Problems that arise during the execution of the programs on the CRAY computer will be documented at the end of the CPR-file before the accounting section. For details on the error message, one may wish to read the description for the given error number in the CRAY operating system (COS) message manual.

Problems that arrise from the use of the VAX are usually indicated by on-the-screen error messages that will indicate the nature of the problem.

If RAM2D1 compiles and runs without any apparent error, then one has no indication that the datafile is incorrect. If then PRAM1 also compiles and runs fine without any apparent error but fails to produce a graph, or produces some graphs as expected but others not at all or only in part, then one should first check the input data to PRAM1, especially the elements of the array `CSEC`. Another suspicion should be that the wrong dataset on the CRAY disk was used. Hence, one should check the existence of the desired datafile, the dimensions, date, and edition number of the file as specified in the input data file, the input namelist, and the program. If all looks well, one can rerun PRAM1, but requesting only one of those graphs that did not come out right previously to narrow the possible sources of error.

If PRAM1 ran without producing a `PLT2.DAT` file the CPR-file might report: `SY001 - RLS COULD NOT FIND A DNT FOR META`, which indicates that the META-file (=DISSPLA terminology for the device independent graphics file PLT2.DAT) was not created or was created, remained empty, and was as such discarded, and hence unavailable for transfer. It may also turn out that the file was held back on the CRAY since there was no room in the VAX directory. Confirm the latter by typing the command `SHOW QUOTA` and delete old files if necessary.

If the execution of the program seems to take an unexpected amount of time, it is likely to be due to the general overload of the computer system or the network rather than due to a problem with the codes. To inquire the computer system performance use the commands presented in section IV.B.3. In addition to those commands, one can check if the submitted process is being worked on which is reflected in an increase in CPU time used. The VAX CPU time can be monitored at any time by typing ^ T (=*Ctrl* T). Caution must be exercised that the *T*-key is hit and not, by accident, the

Y-key, which would terminate the execution. Anagolougously, the CRAY CPU time is listed when monitoring the status of one's own CRAY job by typing `CRAY STATUS/OWN` (`CSO`). Both, `^ T` on the VAX and `CSO` on the CRAY indicate what the computer is currently doing.

PART IV.D CRAY RUN SPECIFICATIONS

The following contains the vital statistics of RAM2D1C examples. The columns are numbered and contain the following information: (recall 1 CRAY word = 8 *bytes* = 64 *bits*)

1. encrypted dimensions
2. time dimension (NT)
3. transverse space dimension (NY)
4. maximum job size when executing (*mega-words*)
5. time executing in CPU for first z-step, 2 data drops (*seconds*)
6. time executing in CPU for 2000 z-steps, 2 data drops in 1-D, 21 data drops in 2-D (*seconds*)
7. CPU time required for compilation (*seconds*)
8. maximum job size when compiling (*mega-words*)
9. size of typical output data file with 2 data drops in 1-D, and 1 data set in 2-D (*mega-words*)

RAM2D1C

1.	2.	3.	4.	5.	6.	7.	8.	9.
IG	512	128	1.40	.730	799.75	2.23	1.46	.79
HH	256	256	1.40	.712	852.00	2.21	1.46	.79
HG	256	128	.74	.333	405.77	2.14	.81	.39
GG	128	128	.41	.158	201.67	2.01	.48	.20
K1	2048	1	.12	.017	12.08	2.06	.20	.037
J1	1024	1	.10	.009	6.11	2.07	.17	.024
I1	512	1	.10	.006	3.11	2.06	.16	.018
H1	256	1	.10	.004	1.63	2.06	.15	.015
1K	1	2048	.14	.062	28.63	2.11	.22	.061
1J	1	1024	.11	.030	14.93	2.17	.18	.037
1I	1	512	.10	.017	7.39	2.18	.16	.024
1H	1	256	.10	.010	3.60	2.15	.15	.018

The following contains the vital statistics of PRAM1CD examples. The columns are numbered and contain the following information:
(recall 1 CRAY word = 8 *bytes* = 64 *bits*
1 VAX block = 512 *bytes*

1. encrypted dimensions
2. time dimension (NT)
3. transverse space dimension (NY)
4. maximum job size when executing (*mega-words*)
5. time executing in CPU for one graph (*seconds*)
6. time executing in CPU for 10 graphs (*seconds*)
7. CPU time required foı compilation (*seconds*)
8. maximum job size when compiling (*mega-words*)
9. size of typical PLT2.DAT file with 1 graph (*blocks*)
10. size of typical PLT2.DAT file with 10 graphs (*blocks*)

PRAM1CD

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
IG	512	128	.60	1.917	11.33	5.917	.675	46	402
HH	256	256	.60	1.279	8.88	5.596	.672	32	288
HG	256	128	.40	.827	6.80	5.63	.46	37	288
GG	128	128	.29	.523	3.78	5.504	.36	26	206
K1	2048	1	.25	.355	3.08	9.20	.34	27	240
J1	1024	1	.22	.253	2.23	9.23	.30	20	177
I1	512	1	.20	.116	1.74	9.24	.28	16	147
H1	256	1	.19	.162	1.43	9.19	.27	13	130
1K	1	2048	.26	.345	2.70	8.699	.34	27	219
1J	1	1024	.22	.257	2.17	9.223	.30	19	164
1I	1	512	.20	.197	1.65	9.20	.28	15	133
1H	1	256	.19	.165	1.56	8.81	.27	13	120

APPENDICES

Appendix A

The appendices A-C present five examples of what the typical input to and output from RAM2D1 and PRAM1 looks like. The input data files, N---.DAT, must not contain any character in the first column of any line! (This is not visible in the examples shown.) All characters start in column 2 and/or the following columns. The input data are grouped in so called **namelists** (variables between two consecutive $-signs. The character strings following the first $-sign is the name of the namelist. The complete list of variables of each namelist is evident from the code listings. The possible values for these variables and the implications of these values are explained there in the commentary preceeding the routine (or subroutine) where the variable is used. For brevity's sake, four pages of the PLT2.DAT graphics output file are reproduced on a single page here. Even this reduction of volume was insufficient in Example B2. Hence, example B2 shows only a choice of the plots that result from the given input data.

APPENDIX A 1-D Transient Limit; Examples

Two examples are appended to show code operation in the transient limit. The illustration features the batch job command files, the input data files, the ouput CPR-files and the resulting output. *The first example is a run that illustrates the* basic use of the codes without complications or finess. The second example illustrates how several one-dimensional simulations can be done while running the programs only once.

EXAMPLE A1

XRJ1.JOB

```
AUDIT.
ACCESS,  DN=XRJ1.
FETCH,   DN=NRAM,TEXT='NRJ1.DAT'.
XRJ1.
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='XRJ1.MSG.'.
AUDIT.
EXIT.
```

XPJ1.JOB

```
AUDIT.
ACCESS,   DN=XPJ1.
ACCESS,   DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=DVSD,ID=DISSPLA,OWN=LIBRARY.
FETCH,    DN=NPRAM1,TEXT='NPJ1.DAT'.
XPJ1.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='XPJ1.MSG.'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='XPJ1.DSP.'.
AUDIT.
EXIT.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='XPJ1.MSG.'.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='XPJ1.DSP.'.
DUMPJOB.
DEBUG,    BLOCKS=GRAPHS.
```

```
AUDIT.
ACCESS,   DN=XPJ1.
ACCESS,   DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=DVSD,ID=DISSPLA,OWN=LIBRARY.
FETCH,    DN=NPRAM1,TEXT='NPJ1.DAT'.
XPJ1.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='XPJ1.MSG.'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='XPJ1.DSP.'.
AUDIT.
EXIT.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='XPJ1.MSG.'.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='XPJ1.DSP.'.
DUMPJOB.
DEBUG,    BLOCKS=GRAPHS.
```

NRJ1.DAT

```
$NAML
  RINT(1)=1.0,
  RIST=1.0E-8,
  ICOND=3,
  ZFINAL=100.0,
  ZKEEP=50.0,
$
     NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
    1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
    2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
```

NPJ1.DAT

```
$FLDATE
  DONYET=1,
  MONTH=04,
  DAY=18,
  YEAR=88,
  IPART=1,
  NEDN=1,
$
$CONDAT
  LPRMT(1)=1,
  LPRMT(2)=1,
  LPRMT(3)=1,
  LPRMT(4)=1,
  NSEC=1,
  CSEC(1,1)=(1.0,2.0),
  CSEC(2,1)=(1.0,2.0),
  CSEC(3,1)=(1.0,2.0),
  CSEC(7,1)=(1.0,2.0),
  CSEC(8,1)=(1.0,2.0),
  CSEC(9,1)=(1.0,2.0),
  CSEC(13,1)=(1.0,2.0),
  CSEC(14,1)=(1.0,2.0),
  CSEC(15,1)=(1.0,2.0),
$
$ZPLOT
  KZ(1)=1,
  KZ(2)=2,
  KZ(3)=3,
$
$RMPLT
$
```

XRJ1.CPR

```
09:02:06 0512    0.0000    CSP    ..........................................................................
09:02:06 0515    0.0000    CSP    '                                                                        '
09:02:06 0518    0.0000    CSP    '                    WELCOME TO THE NRL CRAY XMP                         '
09:02:06.0521    0.0000    CSP    '                                                                        '
09:02:06 0523    0.0001    CSP    ..........................................................................
09:02:06 0526    0.0001    CSP    ' The CRAY will be unavailable Sunday April 24 from 8:00 A.M. to 4:00 P.M '
09:02:06 0529    0.0001    CSP    ' for software testing                                                   '
09:02:06 0531    0.0001    CSP    ..........................................................................
09:02:06 0534    0.0001    CSP    '  There will be no CRAY off-line data set recalls on Tuesday or Wednesday '
09:02:06.0537    0.0001    CSP    '  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP '
09:02:06 0539    0.0001    CSP    '  runs on our CRAY archive tape library.                                '
09:02:06 0542    0.0002    CSP    ..........................................................................
09:02:06 0564    0.0002    CSP          CRAY X-MP SERIAL-415 65     NAVAL RESEARCH LABORATORY     04 21 88
09:02:06 0566    0.0002    CSP
09:02:06 0570    0.0002    CSP          CRAY OPERATING SYSTEM            COS 1.15  ASSEMBLY DATE 01 04 88
09:02:06.0573    0.0002    CSP
09:02:06 0575    0.0002    CSP
09:02:06 3365    0.0002    CSP     JOB.JN=XRJ1.MFL=511000.US=DEFER.
09:02:06 5042    0.0012    CSP     ACCOUNT.AC=.US=.UPW=.APW=
09:02:07 5422    0.1095    USER   AC213    '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
09:02:07 8360    0.1126    USER   AUDIT.
09:02:22.6940    0.3453    USER   AU003 -       213 DATASETS.   226201 BLOCKS.   115746098 WORDS
09:02:22.6944    0.3454    USER   AU003 -        63 DATASETS.    46310 BLOCKS.    23694963 WORDS ONLINE
09:02:22.6949    0.3455    USER   AU003 -       150 DATASETS.   179891 BLOCKS.    92051135 WORDS OFFLINE
09:02:22.7022    0.3457    CSP    ACCESS. DN=XRJ1.
09:02:22 9635    0.3457    PDM    PD000 - PDN = XRJ1            ID =            ED =    5  OWN = HILFER
09:02:22 9637    0.3457    PDM    PD000 - ACCESS   COMPLETE
09:02:22.9652    0.3458    CSP    FETCH.  DN=NRAM.TEXT='NRJ1.DAT'.
09:02:28 3559    0.3459    SCP     VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
09:02:28 3562    0.3459    SCP     VAX TO CRAY: FILE=$1$DUA107:[HILFER.FR2]NRJ1.DAT:21
09:02:28 3564    0.3459    SCP     VAX TO CRAY: 416 BYTES TRANSFERRED
09:02:30 9535    0.3459    SCP    SS004 - DATASET RECEIVED FROM FRONT END
09:02:31 2485    0.3461    CSP    XRJ1
09:02:46.8580   13.0298    PDM    PD000 - PDN = FK1042188        ID =            ED =    1  OWN = HILFER
09:02:46 8582   13.0298    PDM    PD000 - SAVE     COMPLETE
09:02:46.8683   13.0298    USER   UT003 - EXIT CALLED BY  RAM2D1C
09:02:46 8718   13.0299    CSP    DISPOSE. DN=ERRM.DF=BB.WAIT.TEXT='XRJ1.MSG.'.
09:02:53 6929   13.0301    SCP     CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
09:02:53 6932   13.0301    SCP     CRAY TO VAX: FILE=$1$DUA107:[HILFER.FR2]XRJ1.MSG:1
09:02:53.6935   13.0301    SCP     CRAY TO VAX: 20 BYTES TRANSFERRED
09:02:57 5904   13.0305    USER   AUDIT.
09:03:08 6528   13.2640    USER   AU003 -       214 DATASETS.   226297 BLOCKS.   115795201 WORDS
09:03:08 6532   13.2641    USER   AU003 -        64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE
09:03:08 6536   13.2642    USER   AU003 -       150 DATASETS.   179891 BLOCKS.    92051135 WORDS OFFLINE
09:03:08 6599   13.2642    CSP    EXIT.
09:03:08.6617   13.2643    CSP    END OF JOB
09:03:08 6619   13.2643    CSP
09:03:08 6622   13.2643    CSP
09:03:08 8352   13.2644    USER        JOB NAME -                           XRJ1
09:03:08 8355   13.2644    USER        USER NUMBER -                        HILFER
09:03:08 8359   13.2644    USER        JOB SEQUENCE NUMBER -                     40324
09:03:08.8362   13.2644    USER
09:03:08.8367   13.2645    USER        TIME EXECUTING IN CPU -              0000:00:13.2644
09:03:08.8370   13.2645    USER        TIME WAITING TO EXECUTE -            0000:00:26.4189
09:03:08 8373   13.2645    USER        TIME WAITING FOR I/O -               0000:00:22.0685
09:03:08 8376   13.2645    USER        TIME WAITING IN INPUT QUEUE -        0000:00:00.2267
09:03:08 8379   13.2645    USER        MEMORY * CPU TIME (MWDS*SEC) -                1.62992
09:03:08.8383   13.2645    USER        MEMORY * I/O WAIT TIME (MWDS*SEC) -           2.17570
09:03:08.8386   13.2646    USER        MINIMUM JOB SIZE (WORDS) -              44544
09:03:08.8389   13.2646    USER        MAXIMUM JOB SIZE (WORDS) -             124416
```

XRJ1.CPR

```
09:03:08.8392     13.2646    USER        MINIMUM FL (WORDS)                        40960
09:03:08.8395     13.2646    USER        MAXIMUM FL (WORDS) -                     119808
09:03:08.8398     13.2646    USER        MINIMUM JTA (WORDS) -                      3584
09:03:08.8401     13.2646    USER        MAXIMUM JTA (WORDS) -                      4608
09:03:08.8405     13.2646    USER        DISK SECTORS MOVED -                       2302
09:03:08.8408     13.2646    USER        FSS SECTORS MOVED -                           0
09:03:08.8411     13.2646    USER        USER I O REQUESTS -                        1397
09:03:08.8414     13.2646    USER        USER I O SUSPENSIONS -                     1544
09:03:08.8417     13.2646    USER        OPEN CALLS -                                 27
09:03:08.8421     13.2647    USER        CLOSE CALLS -                                28
09:03:08.8424     13.2647    USER        MEMORY RESIDENT DATASETS -                    0
09:03:08.8427     13.2647    USER        TEMPORARY DATASET SECTORS USED -              1
09:03:08.8430     13.2647    USER        PERMANENT DATASET SECTORS ACCESSED -       1600
09:03:08.8434     13.2647    USER        PERMANENT DATASET SECTORS SAVED -            98
09:03:08.8437     13.2647    USER        SECTORS RECEIVED FROM FRONT END -             1
09:03:08.8440     13.2647    USER        SECTORS QUEUED TO FRONT END -                 1
09:03:09.1518     13.2724    USER
09:03:09.1520     13.2724    USER    ..........................................................................
09:03:09.1524     13.2725    USER                          '''  COST TABLE FOR THIS JOB '''
09:03:09.1527     13.2725    USER                      JOBNAME -----------                            XRJ1
09:03:09.1531     13.2726    USER                      USER IDENT.---------                           HILFER
09:03:09.1534     13.2727    USER                      BEGAN EXECUTION ----  THU APR 21, 1988         09:02:05  HOURS
09:03:09.1578     13.2728    USER                      AT A PRIORITY OF  --                              · 3
09:03:09.1582     13.2729    USER                      AND JOB CLASS OF  --                           DSMALL
09:03:09.1585     13.2730    USER            13.271129 SECONDS OF CPU TIME      @ $ 630.00  HR       --    $        2.32
09:03:09.1589     13.2732    USER             1.630306 MEMORY*CPU (MWRD-SEC)    @ $  84.00  HR       --    $        0.04
09:03:09.1593     13.2733    USER             2.177428 MEMORY*I O (MWRD-SEC)    @ $  84.00  HR       --    $        0.05
09:03:09.1597     13.2734    USER             0.002303 I O MEGASECTORS MOVED    @ $  84.00  EA       --    $        0.19
09:03:09.1600     13.2735    USER             0.000000 TAPE MOUNT(S)            @ $   5.00  EA       --    $        0.00
09:03:09.1604     13.2736    USER
09:03:09.1606     13.2736    USER                      ''''    TOTAL COST FOR THIS JOB     ''''     --    $        2.60
09:03:09.1609     13.2736    USER    ..........................................................................
```

XPJ1.CPR

```
09:03:27.5941     0.0000   CSP    ***************************************************************************
09:03:27.5944     0.0000   CSP    *                                                                         *
09:03:27.5947     0.0000   CSP    *                    WELCOME TO THE NRL CRAY XMP                          *
09:03:27.5950     0.0001   CSP    *                                                                         *
09:03:27.5953     0.0001   CSP    ***************************************************************************
09:03:27.5955     0.0001   CSP    * The CRAY will be unavailable Sunday April 24 from 8:00 A.M. to 4:00 P.M. *
09:03:27.5958     0.0001   CSP    * for software testing.                                                   *
09:03:27.5961     0.0001   CSP    ***************************************************************************
09:03:27.5963     0.0001   CSP    *  There will be no CRAY off-line data set recalls on Tuesday or Wednesday *
09:03:27.5966     0.0001   CSP    *  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP *
09:03:27.5969     0.0001   CSP    *  runs on our CRAY archive tape library.                                 *
09:03:27.5972     0.0002   CSP    ***************************************************************************
09:03:27.5993     0.0002   CSP           CRAY X-MP SERIAL-415.65    NAVAL RESEARCH LABORATORY    04 21 88
09:03:27.5996     0.0002   CSP
09:03:27.5999     0.0002   CSP           CRAY OPERATING SYSTEM          COS 1.15  ASSEMBLY DATE 01 04 88
09:03:27.6002     0.0002   CSP
09:03:27.6004     0.0002   CSP
09:03:27.6122     0.0002   CSP     JOB.JN-XPJ1.MFL-511000.US-DEFER.
09:03:27.6476     0.0014   CSP     ACCOUNT,AC-.US-.UPW-.APW-.
09:03:28.7831     0.1099   USER   AC213 - '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
09:03:29.0537     0.1131   USER   AUDIT.
09:03:40.1793     0.3464   USER   AU003 -      214 DATASETS.   226297 BLOCKS.   115795201 WORDS
09:03:40.1797     0.3465   USER   AU003 -       64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE
09:03:40.1801     0.3466   USER   AU003 -      150 DATASETS.   179891 BLOCKS.    92051135 WORDS OFFLINE
09:03:40.1875     0.3468   CSP    ACCESS.  DN-XPJ1.
09:03:40.4632     0.3468   PDM    PD000 - PDN - XPJ1             ID -             ED -   39  OWN - HILFER
09:03:40.4634     0.3468   PDM    PD000 - ACCESS  COMPLETE
09:03:40.4652     0.3472   CSP    ACCESS.  DN-DISLIB.ID-DISSPLA.OWN-LIBRARY.
09:03:40.7388     0.3472   PDM    PD000 - PDN - DISLIB           ID - DISSPLA     ED -    1  OWN - LIBRARY
09:03:40.7390     0.3472   PDM    PD000 - ACCESS  COMPLETE
09:03:40.7408     0.3476   CSP    ACCESS.  DN-INTLIB.ID-DISSPLA.OWN-LIBRARY.
09:03:40.9784     0.3476   PDM    PD000 - PDN - INTLIB           ID - DISSPLA     ED -    1  OWN - LIBRARY
09:03:40.9787     0.3476   PDM    PD000 - ACCESS  COMPLETE
09:03:40.9805     0.3479   CSP    ACCESS.  DN-DVSD.ID-DISSPLA.OWN-LIBRARY.
09:03:41.2152     0.3480   PDM    PD000 - PDN - DVSD             ID - DISSPLA     ED -    1  OWN - LIBRARY
09:03:41.2154     0.3480   PDM    PD000 - ACCESS  COMPLETE
09:03:41.2170     0.3480   CSP    FETCH.   DN-NPRAM1.TEXT-'NPJ1.DAT'.
09:03:43.0104     0.3482   SCP     VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
09:03:43.0107     0.3482   SCP     VAX TO CRAY: FILE-$1$DUA107:[HILFER.FR2]NPJ1.DAT;39
09:03:43.0111     0.3482   SCP     VAX TO CRAY: 768 BYTES TRANSFERRED
09:03:47.1131     0.3482   SCP    SS004 - DATASET RECEIVED FROM FRONT END
09:03:47.3965     0.3483   CSP    XPJ1.
09:03:47.9514     0.3516   PDM    PD000 - PDN - FK1042188        ID -             ED -    1  OWN - HILFER
09:03:47.9516     0.3516   PDM    PD000 - ACCESS  COMPLETE
09:04:02.1912    13.5244   USER   UT003 - EXIT CALLED BY  PRAM1CD
09:04:02.1938    13.5244   CSP    DISPOSE, DN-META.DF-BB.WAIT.TEXT-'PLT2 DAT'
09:04:17.8375    13.5247   SCP     CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
09:04:17.8378    13.5247   SCP     CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]PLT2.DAT;2
09:04:17.8381    13.5247   SCP     CRAY TO VAX: 496240 BYTES TRANSFERRED
09:04:24.4376    13.5247   CSP    DISPOSE, DN-EPRM.DF-BB.WAIT.TEXT-'XPJ1.MSG.'.
09:04:29.4986    13.5249   SCP     CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
09:04:29.4988    13.5249   SCP     CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]XPJ1.MSG;1
09:04:29.4991    13.5249   SCP     CRAY TO VAX: 3101 BYTES TRANSFERRED
09:04:33.7486    13.5250   CSP    DISPOSE, DN-DISOUT.DF-BB.WAIT.TEXT-'XPJ1.DSP.'.
09:04:38.5871    13.5252   SCP     CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
09:04:38.5874    13.5252   SCP     CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]XPJ1.DSP;1
09:04:38.5877    13.5252   SCP     CRAY TO VAX: 888 BYTES TRANSFERRED
09:04:43.1980    13.5256   USER   AUDIT.
09:04:54.3707    13.7600   USER   AU003 -      214 DATASETS.   226297 BLOCKS.   115795201 WORDS
09:04:54.3711    13.7601   USER   AU003 -       64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE
```

XPJ1.CPR

```
09:04:54 3718     13 7603    USER    AU003 -      150 DATASETS,   179891 BLOCKS,    92051135 WORDS OFFLINE
09:04:54 3792     13 7603    CSP     EXIT.
09:04:54 3806     13 7603    CSP     END OF JOB
09:04:54 3809     13 7603    CSP
09:04:54 3811     13 7603    CSP
09:04:54 5244     13 7605    USER
09:04:54 5247     13 7605    USER
09:04:54 5250     13 7605    USER
09:04:54 5253     13 7605    USER
09:04:54 5257     13 7605    USER
09:04:54 5260     13 7605    USER
09:04:54 5263     13 7605    USER
09:04:54 5268     13 7606    USER
09:04:54 5272     13 7606    USER
09:04:54 5275     13 7606    USER
09:04:54 5278     13 7606    USER
09:04:54 5282     13 7606    USER
09:04:54 5285     13 7606    USER
09:04:54 5288     13 7606    USER
09:04:54 5291     13 7607    USER
09:04:54 5294     13 7607    USER
09:04:54 5298     13 7607    USER
09:04:54 5301     13 7607    USER
09:04:54 5373     13 7607    USER
09:04:54 5376     13 7607    USER
09:04:54 5379     13 7607    USER
09:04:54 5382     13 7607    USER
09:04:54 5385     13 7607    USER
09:04:54 5389     13 7607    USER
09:04:54 5392     13 7607    USER
09:04:54 5395     13 7607    USER
09:04:54 5398     13 7608    USER
09:04:54 5401     13 7608    USER
09:04:54 8575     13 7685    USER
09:04:54 8577     13 7685    USER
09:04:54 8581     13 7686    USER
09:04:54 8584     13 7686    USER
09:04:54 8588     13 7687    USER
09:04:54 8591     13 7688    USER
09:04:54 8595     13 7689    USER
09:04:54 8598     13 7690    USER
09:04:54 8602     13 7691    USER
09:04:54 8606     13 7693    USER
09:04:54 8610     13 7694    USER
09:04:54 8613     13 7695    USER
09:04:54 8617     13 7696    USER
09:04:54 8621     13 7697    USER
09:04:54 8623     13 7697    USER
09:04:54 8626     13 7697    USER
```

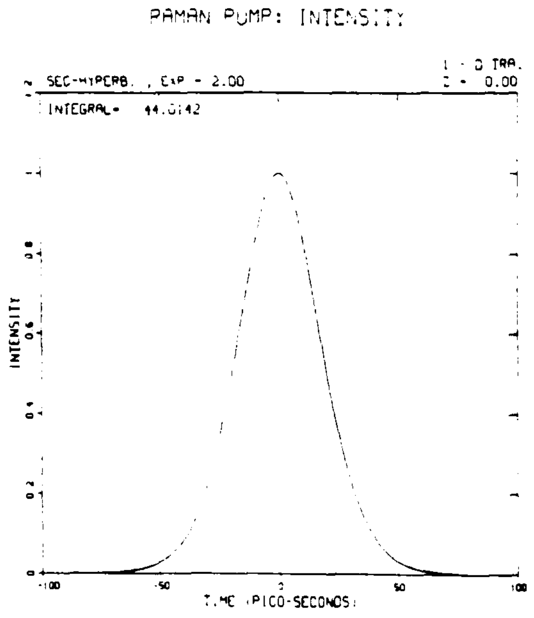
```
JOB NAME -                                      XPJ1
USER NUMBER -                                   HILFER
JOB SEQUENCE NUMBER -                               40330

TIME EXECUTING IN CPU -                         0000:00:13.7605
TIME WAITING TO EXECUTE -                       0000:00:49 1493
TIME WAITING FOR I O -                          0000:00:22 9771
TIME WAITING IN INPUT QUEUE -                   0000:00:00.5499
MEMORY * CPU TIME (MWDS*SEC) -                          3 39919
MEMORY * I O WAIT TIME (MWDS*SEC) -                     2.47661
MINIMUM JOB SIZE (WORDS) -                      44544
MAXIMUM JOB SIZE (WORDS) -                     253952
MINIMUM FL (WORDS) -                            40960
MAXIMUM FL (WORDS) -                           249344
MINIMUM JTA (WORDS) -                            3584
MAXIMUM JTA (WORDS) -                            5120
DISK SECTORS MOVED -                             3072
FSS SECTORS MOVED -                                 0
USER I O REQUESTS -                              1464
USER I O SUSPENSIONS -                           1714
OPEN CALLS -                                       31
CLOSE CALLS -                                      31
MEMORY RESIDENT DATASETS -                          0
TEMPORARY DATASET SECTORS USED -                  127
TEMPORARY DATASET SECTORS ACCESSED -             2821
PERMANENT DATASET SECTORS SAVED -                   0
PERMANENT DATASET SECTORS RECEIVED FROM FRONT END - 1
SECTORS QUEUED TO FRONT END -                     126
...............................................................................................
                 ...   COST TABLE FOR THIS JOB ...
               JOBNAME -----------                                   XPJ1
               USER IDENT ---------                                  HILFER
               BEGAN EXECUTION ----   THU APR 21, 1988               09:03:27  HOURS.
               AT A PRIORITY OF  --                                         3
               AND JOB CLASS OF  --                                  DSMALL
   13.767250   SECONDS OF CPU TIME         @ $ 630.00  HR    --   $        2 41
    3.399609   MEMORY*CPU (MWRD-SEC)       @ $  84.00  HR    --   $        0.08
    2.479123   MEMORY*I O (MWRD-SEC)       @ $  64 00  HR    --   $        0.06
    0.003074   I O MEGASECTORS MOVED       @ $  84.00  EA    --   $        0.26
    0.000000   TAPE MOUNT(S)               @ $   5.00  EA    --   $        0 00
               ....    TOTAL COST FOR THIS JOB            ....  --   $        2 81
...............................................................................................
```

PLT2.DAT (Example A1)

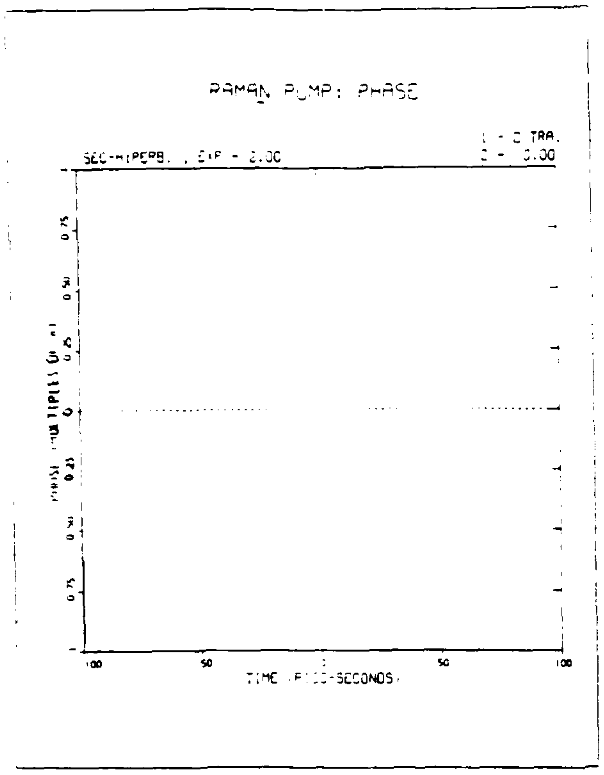
```
                    LIST OF INPUT PARAMETERS

ICOND    -      3
NMAX     -      4000
NPUMP    -      2
NT       -      1024
NY       -      1
GAIN     -       3.0000
PHST     -       0.0000
RALASM   -       5.0000
RAMASM   -       1.5000
RIST     -       1.00*10^4
RKP      -       1.18*10^5
RKS      -       9.19*10^4
TOC      -       5.0000
TOST     -      -40.000
TTWO     -       633.00
TWST     -       40.000
ZFINAL   -       100.00
ZKEEP    -       50.000
ZSTEP    -       0.0500
```

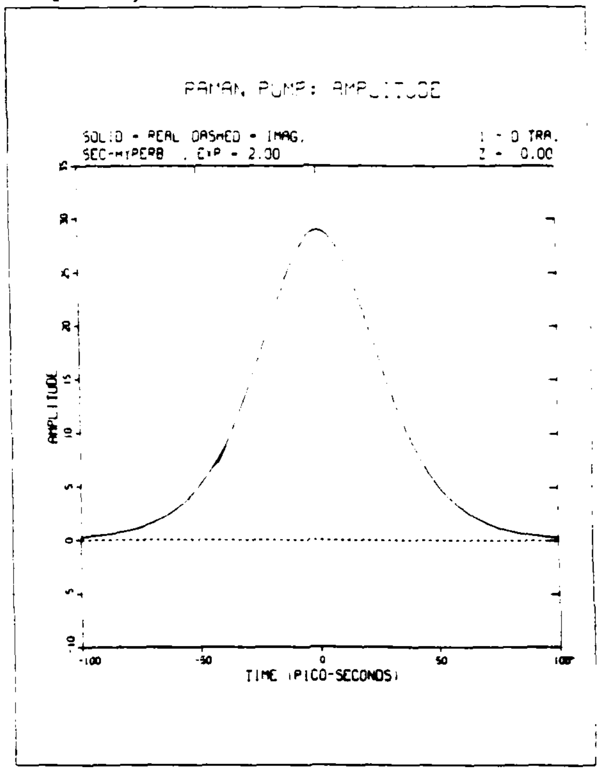
```
LIST OF INPUT PARAMETERS (CONTD)

ITYPE        -     1        1        1        1        1
PHL(1-10)    -   0.0000   0.0000   0.0000   0.0000   0.0000
                 0.0000   0.0000   0.0000   0.0000   0.0000
RINT(1-10)   -   1.0000   0.5500   0.5500   0.5500   0.5500
                 0.5500   0.5500   0.5500   0.5500   0.5500
RTYPE        -   2.0000   2.0000   2.0000   2.0000   2.0000
                 2.0000   2.0000   2.0000
TM(1,2)      -  -100.00   100.00
TOFF(1-10)   -   0.0000   0.0000   0.0000   0.0000   0.0000
                 0.0000   0.0000   0.0000   0.0000   0.0000
TWIDTH       -   40.000   40.000   40.000   40.000   40.000
                 40.000   40.000   40.000   40.000   40.000
```

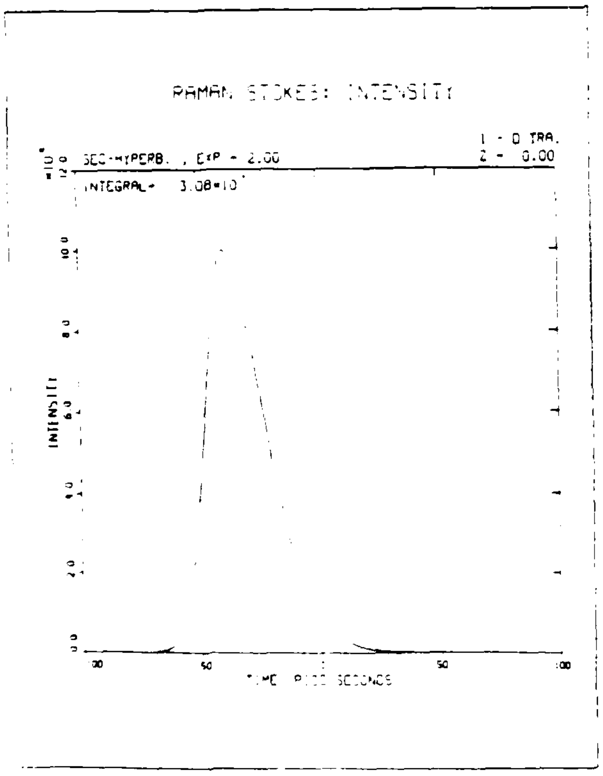
```
LIST OF INPUT PARAMETERS (CONTD)

CSEC(1-19,1-8) -
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
   0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
```

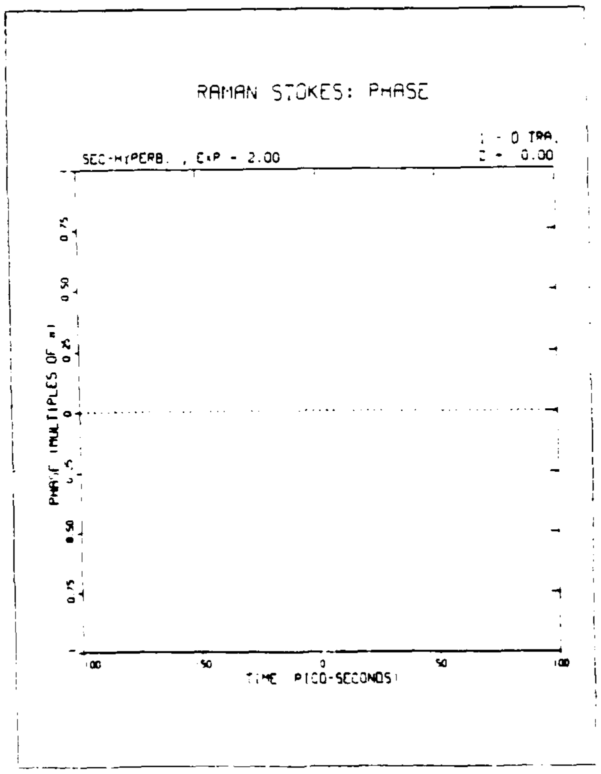


PLT2.DAT (Example A1)

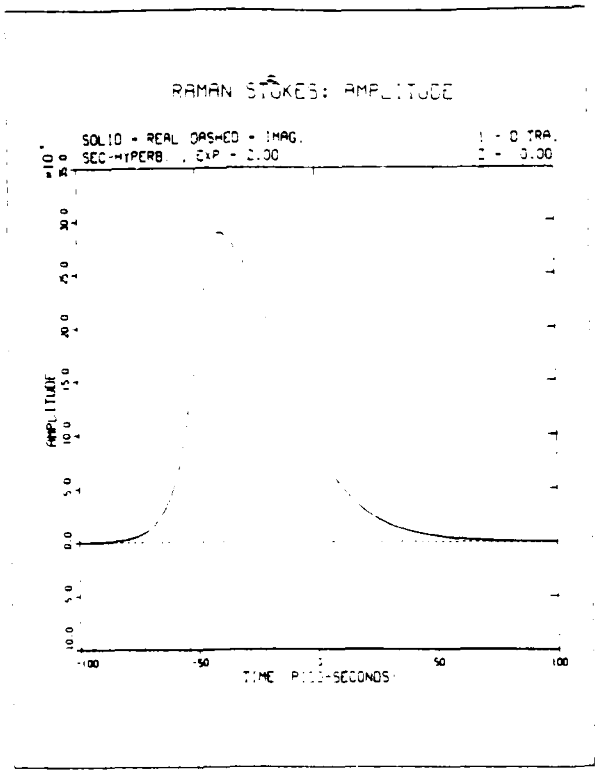


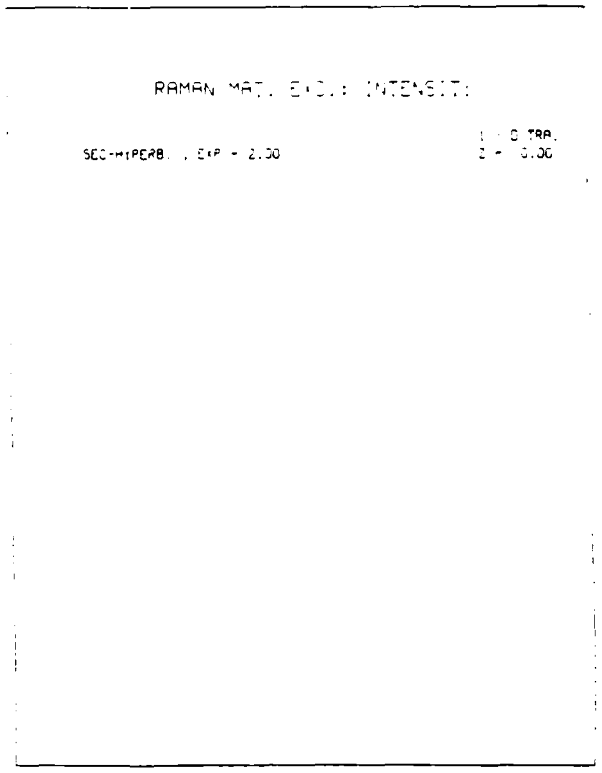


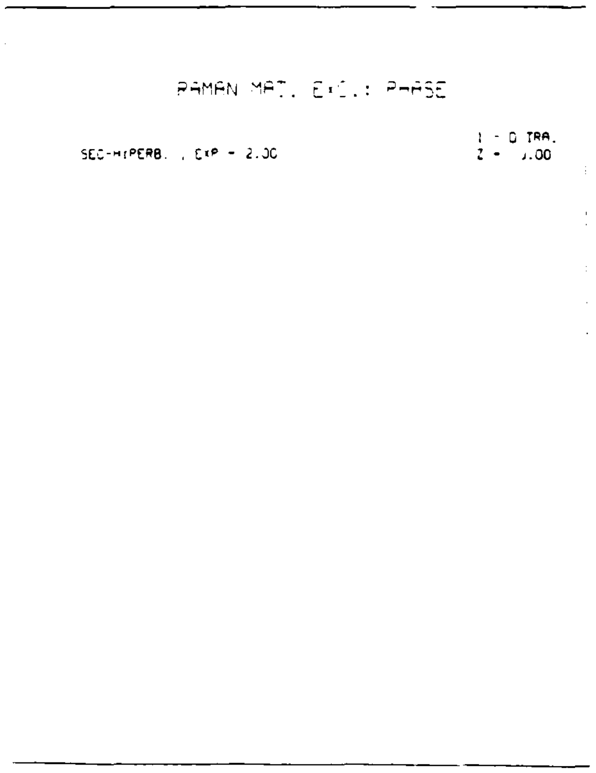


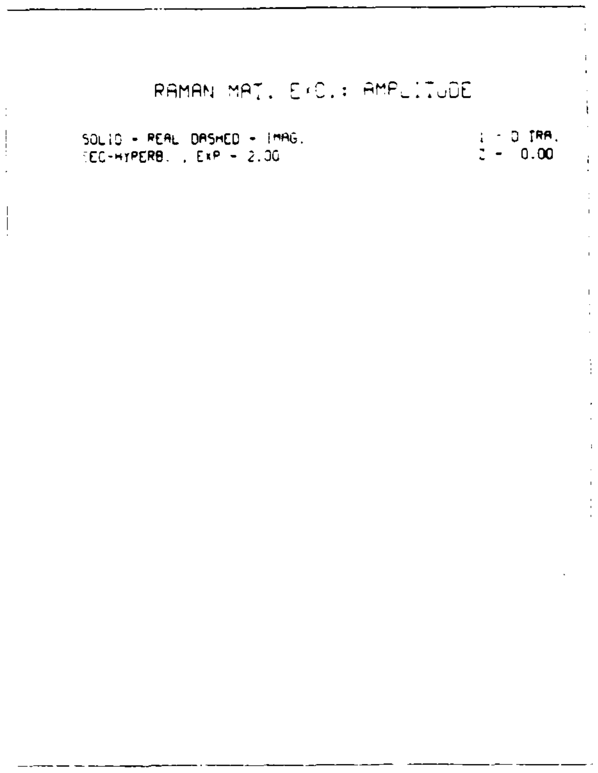


PLT2.DAT (Example A1)

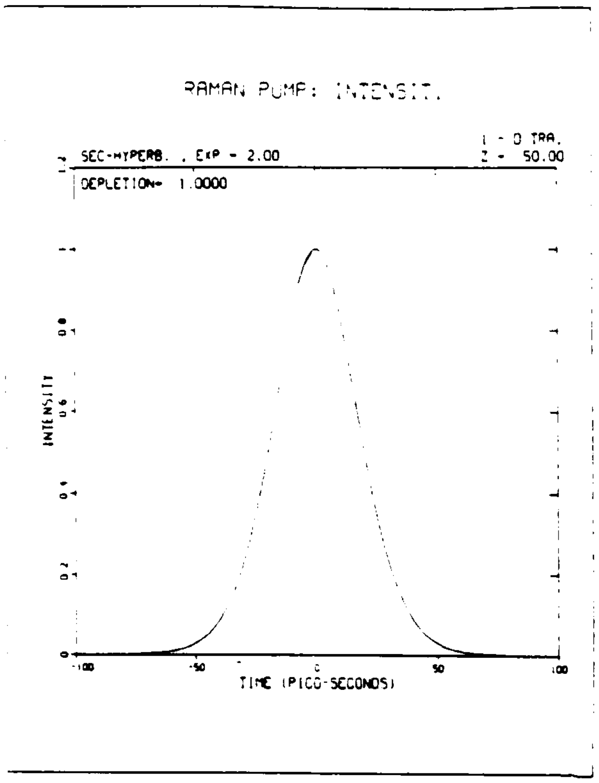


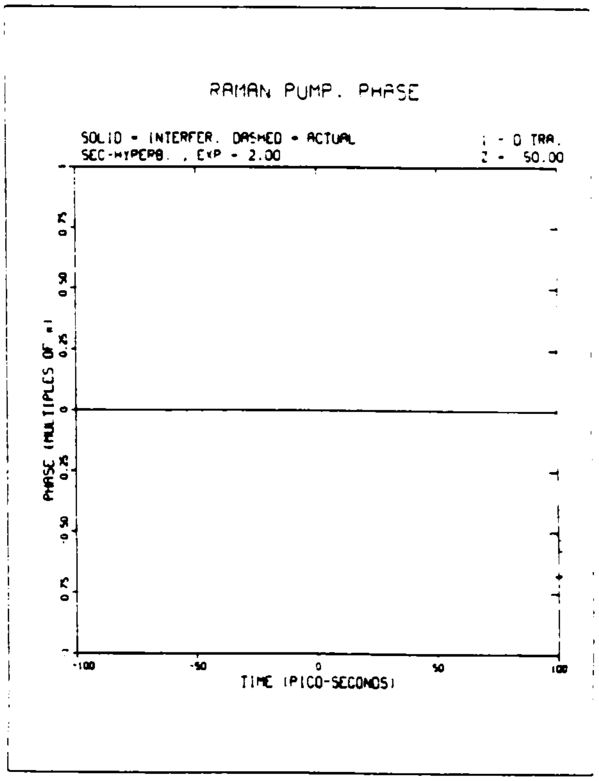


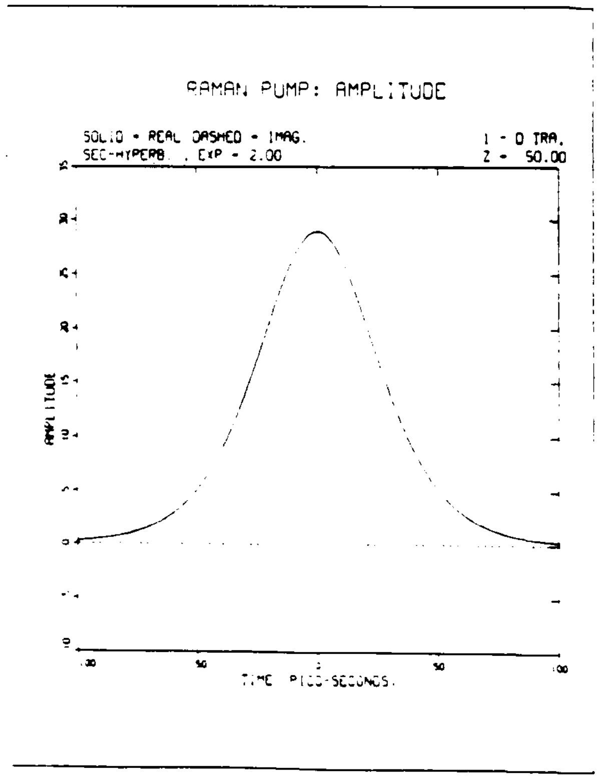


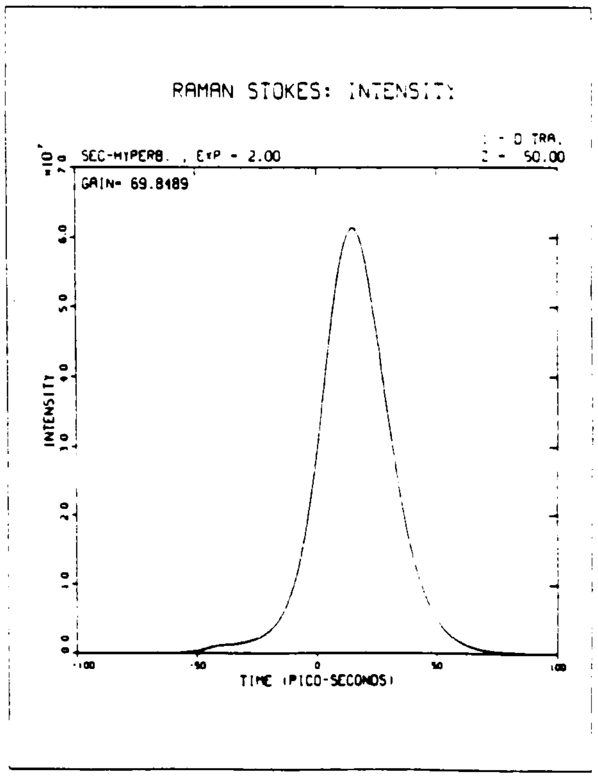


PLT2.DAT (Example A1)

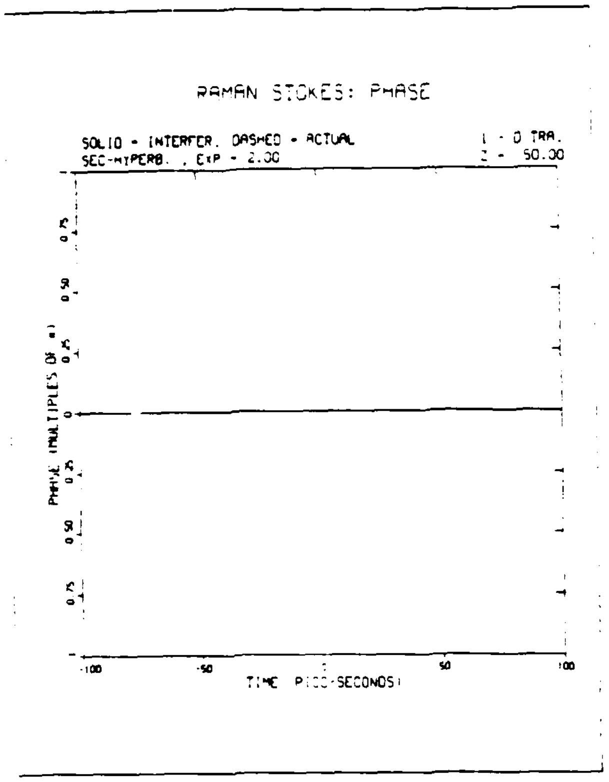


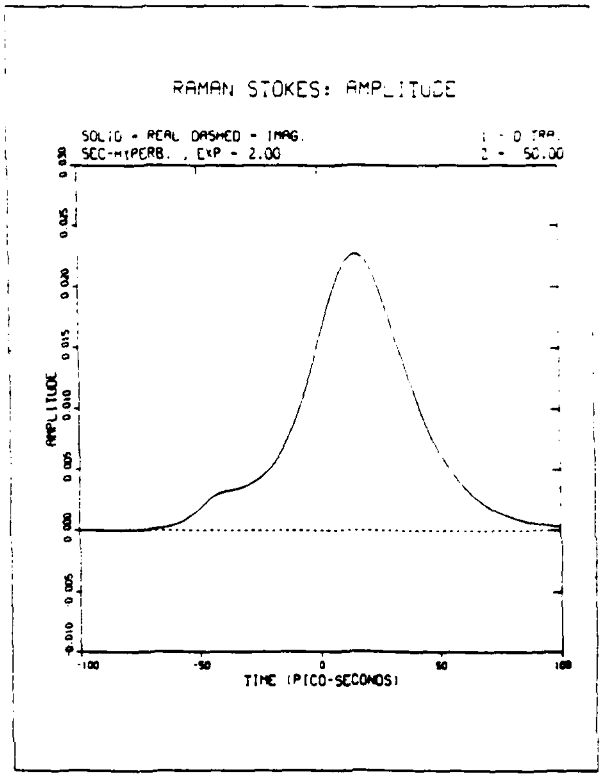


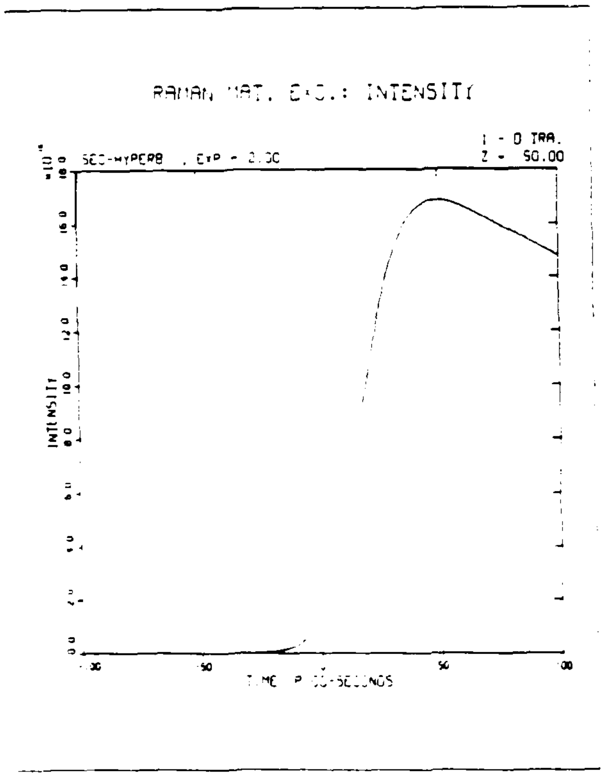


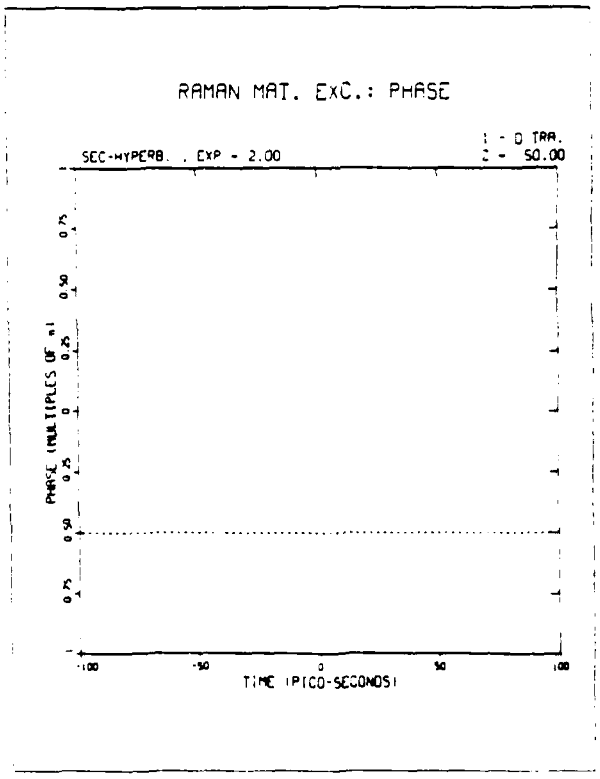


PLT2.DAT (Example A1)

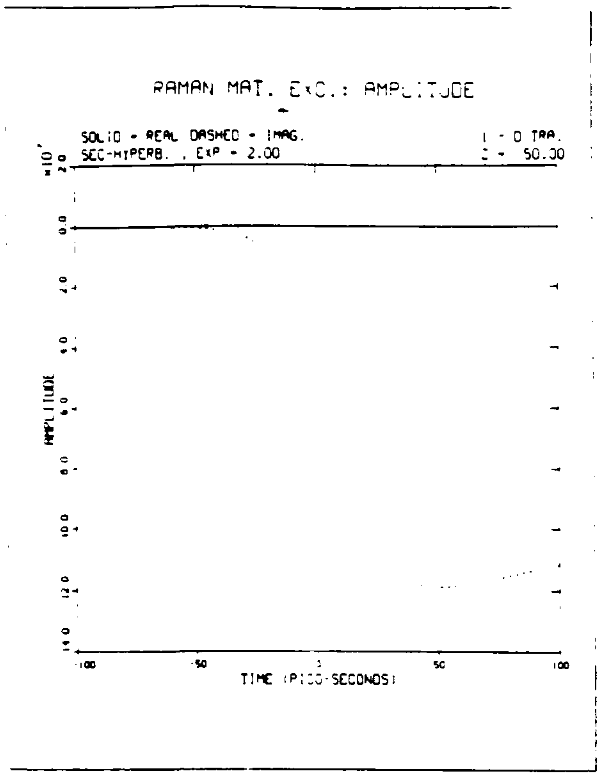


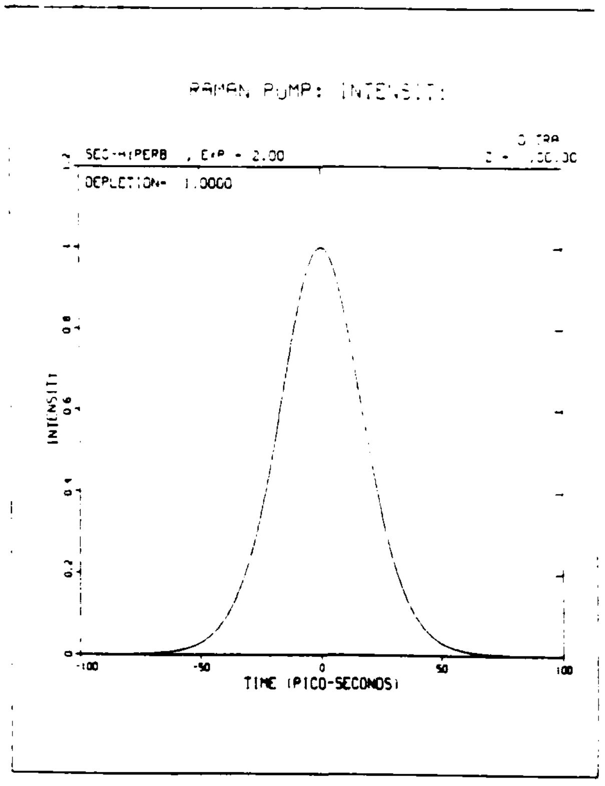


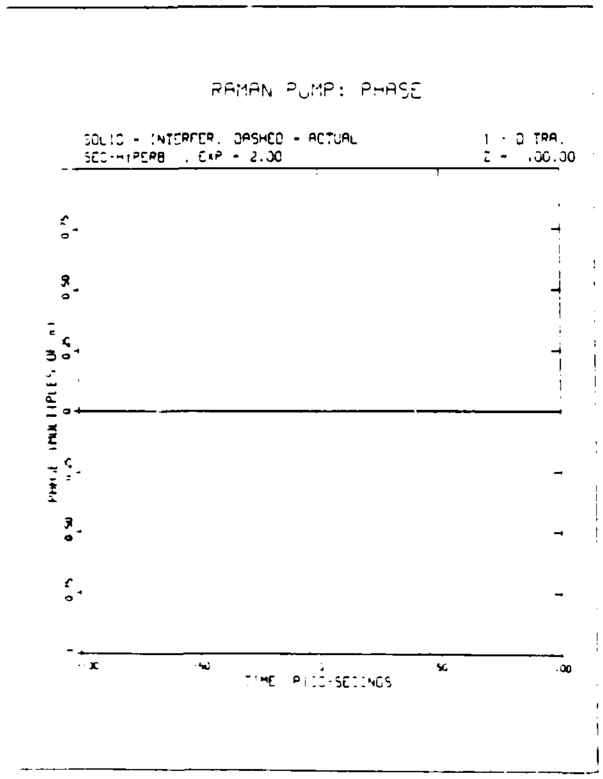


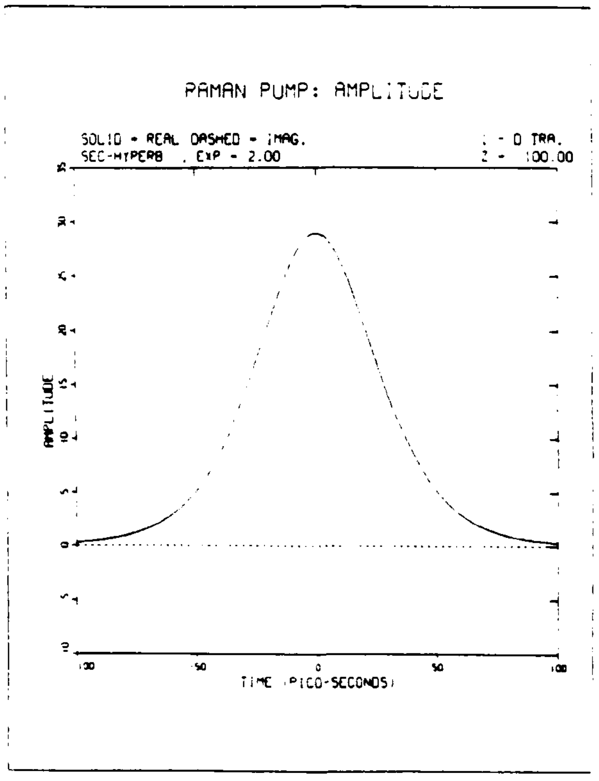


PLT2.DAT (Example A1)

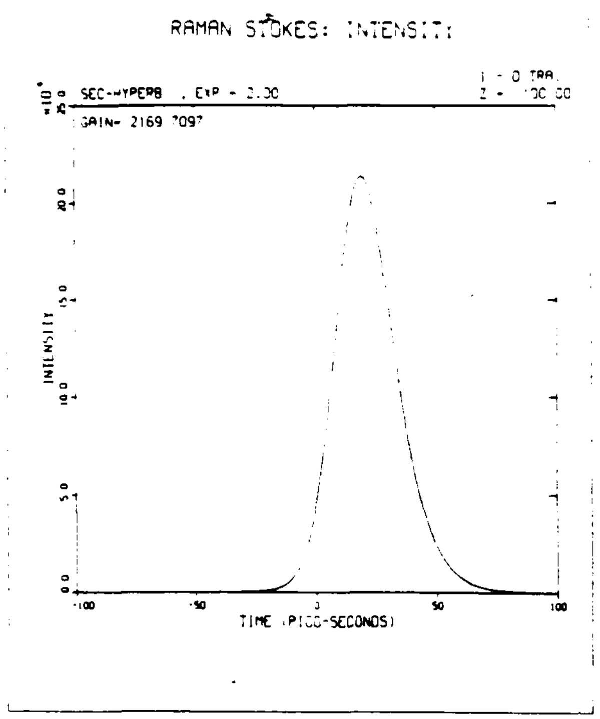


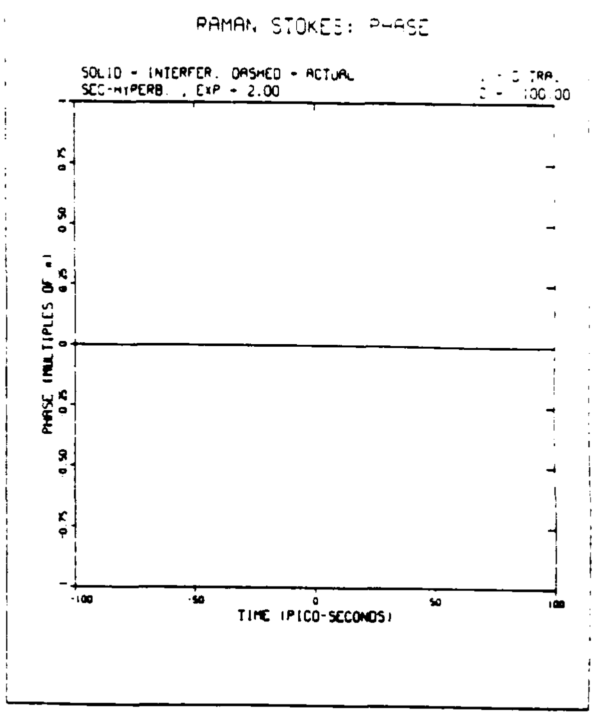


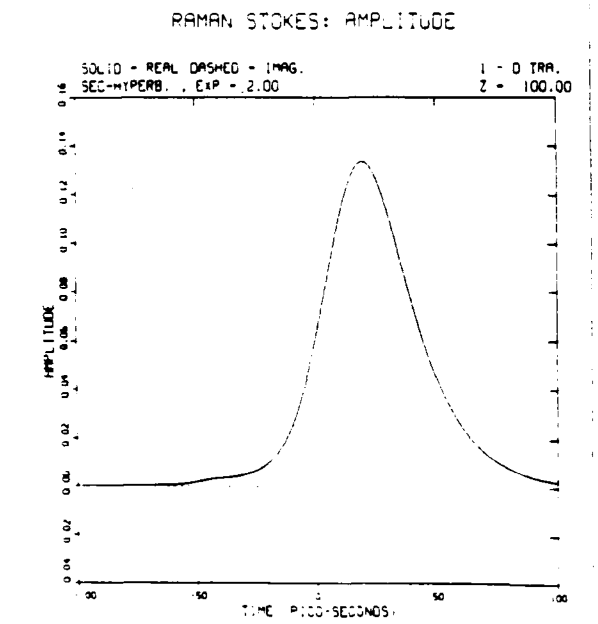


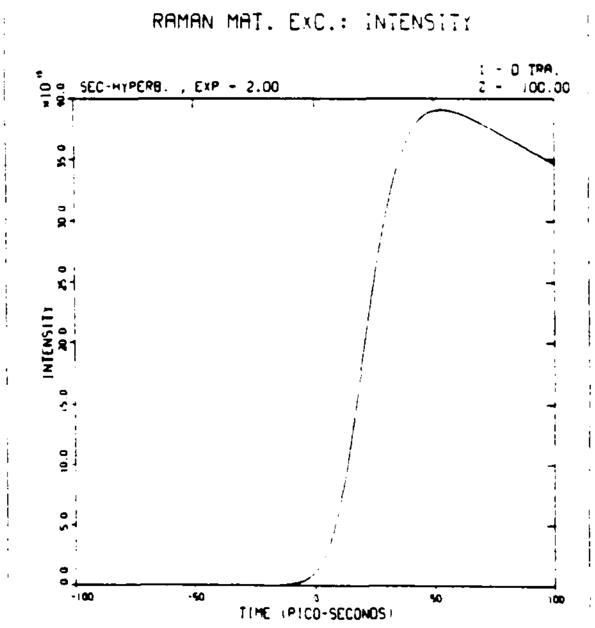


PLT2.DAT (Example A1)

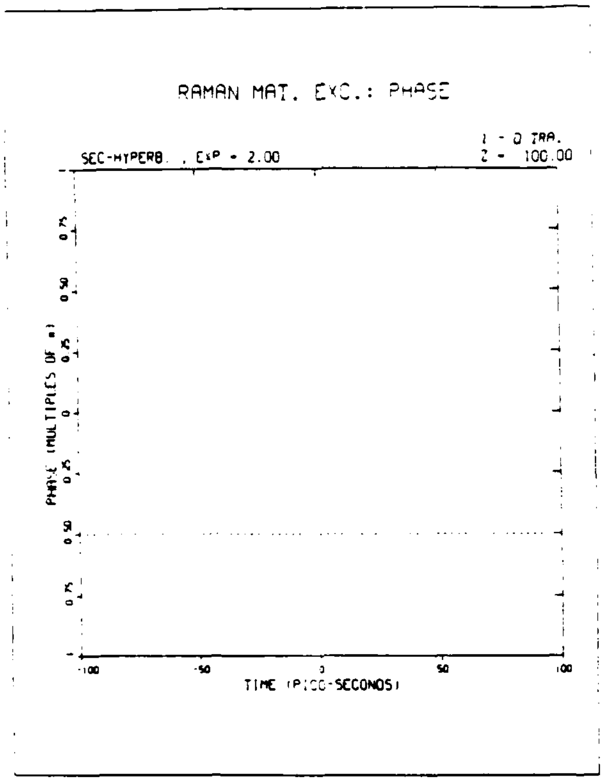


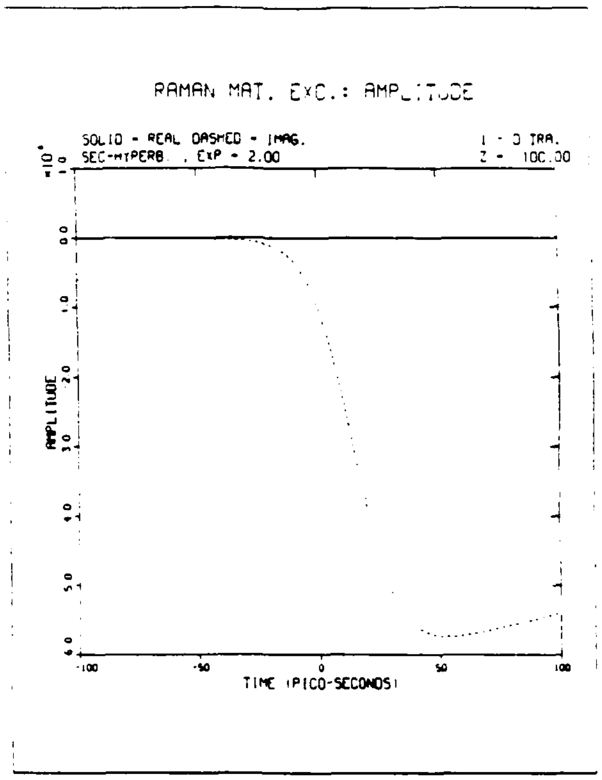






PLT2.DAT (Example A1)





EXAMPLE A2

CRJ3.JOB

```
AUDIT.
FETCH,    DN=RJ3,TEXT='RAM2D1C.FOR'.
CFT,      I=RJ3,ON=INZ.
LDR,      AB=XRJ3,NX.
SAVE,     DN=XRJ3.
FETCH,    DN=NRAM,TEXT='NRJ3.DAT'.
XRJ3.
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='CRJ3.MSG.'.
AUDIT.
EXIT.
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='CRJ3.MSG.'.
DUMPJOB.
DEBUG,    BLOCKS=VINIT.
```

CPJ3.JOB

```
AUDIT.
ACCESS,  DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,  DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,  DN=DVSD,ID=DISSPLA,OWN=LIBRARY.
FETCH,   DN=PJ3,TEXT='PRAM1CD.FOR'.
CFT,     I=PJ3,ON=INZ.
LDR,     LIB=INTLIB:DISLIB,NX,AB=XPJ3.
SAVE,    DN=XPJ3.
RELEASE, DN=PJ3.
FETCH,   DN=NPRAM1,TEXT='NPJ3.DAT'.
XPJ3.
DISPOSE, DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE, DN=EPRM,DF=BB,WAIT,TEXT='CPJ3.MSG.'.
DISPOSE, DN=META,DF=BB,WAIT,TEXT='CPJ3.DSP'.
AUDIT.
EXIT.
DISPOSE, DN=EPRM,DF=BB,WAIT,TEXT='CPJ3.MSG.'.
DISPOSE, DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE, DN=META,DF=BB,WAIT,TEXT='CPJ3.DSP'.
DUMPJOB.
DEBUG,   BLOCKS=GRAPHS.
```

NRJ3.DAT

```
$NAML
  TOFF(2)=30.0,
  TWIDTH(2)=20.0,
  RINT(1)=1.0,
  RINT(2)=10.0,
  RINT(3)=1.0,
  PHL(3)=6.28,
  ITYPE(3)=4,
  RTYPE(3)=8.0,
  RIST=1.0E-8,
  ICOND=3,
  ZFINAL=100.0,
  ZKEEP=50.0,
$
      NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
     1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
     2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
```

NPJ3.DAT

```
$FLDATE
  DONYET=1,
  MONTH=03,
  DAY=28,
  YEAR=88,
  IPART=2,
  NEDN=2,
$
$CONDAT
  LPRMT(1)=1,
  LPRMT(2)=1,
  LPRMT(3)=1,
  LPRMT(4)=1,
  NSEC=3,
  CSEC(1,1)=(1.0,2.0),
  CSEC(1,2)=(2.0,2.0),
  CSEC(1,3)=(3.0,2.0),
  CSEC(2,1)=(1.0,2.0),
  CSEC(2,2)=(2.0,2.0),
  CSEC(2,3)=(3.0,2.0),
  CSEC(7,1)=(1.0,2.0),
  CSEC(7,2)=(2.0,2.0),
  CSEC(7,3)=(3.0,2.0),
  CSEC(8,1)=(1.0,2.0),
  CSEC(8,2)=(2.0,2.0),
  CSEC(8,3)=(3.0,2.0),
  CSEC(13,1)=(1.0,2.0),
  CSEC(13,2)=(2.0,2.0),
  CSEC(13,3)=(3.0,2.0),
  CSEC(14,1)=(1.0,2.0),
  CSEC(14,2)=(2.0,2.0),
  CSEC(14,3)=(3.0,2.0),
$
$ZPLOT
  KZ(1)=1,
  KZ(2)=2,
  KZ(3)=3,
$
```

CRJ3.CPR

```
10:45:04.7223     0.0000   CSP    ...........................................................................
10:45:04.7226     0.0000   CSP    .                                                                         .
10:45:04.7229     0.0000   CSP    .                      WELCOME TO THE NRL CRAY XMP                        .
10:45:04.7232     0.0000   CSP    .                                                                         .
10:45:04.7235     0.0001   CSP    ...........................................................................
10:45:04.7238     0.0001   CSP    .  There will be no CRAY off line data set recalls on Tuesday or Wednesday .
10:45:04.7241     0.0001   CSP    .  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP .
10:45:04.7244     0.0001   CSP    .  runs on our CRAY archive tape library.                                  .
10:45:04.7246     0.0001   CSP    ...........................................................................
10:45:04.9833     0.0001   CSP          CRAY X-MP SERIAL-415 65     NAVAL RESEARCH LABORATORY     04 20 88
10:45:04.9836     0.0001   CSP
10:45:04.9840     0.0001   CSP          CRAY OPERATING SYSTEM           COS 1.15  ASSEMBLY DATE 01 04 88
10:45:04.9843     0.0001   CSP
10:45:04.9846     0.0001   CSP
10:45:05.0688     0.0002   CSP     JOB.JN=CRJ3.MFL=511000.US=DEFER.
10:45:05.7984     0.0012   CSP     ACCOUNT.AC=.US=.UPW=.APW=.
10:45:07.5805     0.1104   USER   AC213 - '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
10:45:08.5479     0.1135   USER   AUDIT
10:45:41.5632     0.3197   USER   AU003 -       187 DATASETS,   208621 BLOCKS,   106751139 WORDS
10:45:41.5637     0.3198   USER   AU003 -        37 DATASETS,    28730 BLOCKS,    14700004 WORDS ONLINE
10:45:41.5642     0.3199   USER   AU003 -       150 DATASETS,   179891 BLOCKS,    92051135 WORDS OFFLINE
10:45:41.5711     0.3200   CSP    FETCH,    DN=RJ3,TEXT='RAM2D1C.FOR'.
10:45:45.7731     0.3201   SCP     VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
10:45:45.7736     0.3201   SCP     VAX TO CRAY: FILE=$1$DUA107:[HILFER.FR2]RAM2D1C.FOR;46
10:45:45.7740     0.3201   SCP     VAX TO CRAY: 54936 BYTES TRANSFERRED
10:45:50.1769     0.3201   SCP    SS004 - DATASET RECEIVED FROM FRONT END
10:45:50.4813     0.3205   CSP    CFT.      I=RJ3,ON=INZ.
10:45:50.9744     0.3209   USER   CF000 - CFT VERSION -    06 16 87 1.15BF2
10:45:58.7978     2.1721   USER   CF001 - COMPILE TIME -     1.8512 SECONDS
10:45:58.7982     2.1721   USER   CF002 -       1295 LINES,       803 STATEMENTS
10:45:58.7988     2.1721   USER   CF003 -     75082 WORDS,     14540 I/O BUFFERS USED
10:45:58.7994     2.1723   USER   CF017 -          2 WARNINGS
10:45:58.9784     2.1727   CSP    LDR,      AB=XRJ3,NX.
10:46:17.2556     2.4336   CSP    SAVE,     DN=XRJ3.
10:46:17.5188     2.4336   PDM    PD000 - PDN - XRJ3              ID -              ED -     5  OWN - HILFER
10:46:17.5191     2.4336   PDM    PD000 - SAVE     COMPLETE
10:46:17.5211     2.4337   CSP    FETCH,    DN=NRAM,TEXT='NRJ3.DAT'.
10:46:21.7099     2.4338   SCP     VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
10:46:21.7102     2.4338   SCP     VAX TO CRAY: FILE=$1$DUA107:[HILFER.FR2]NRJ3.DAT;3
10:46:21.7106     2.4338   SCP     VAX TO CRAY: 592 BYTES TRANSFERRED
10:46:25.3771     2.4338   SCP    SS004 - DATASET RECEIVED FROM FRONT END
10:46:25.6713     2.4340   CSP    XRJ3.
10:47:45.8254    21.3838   PDM    PD000 - PDN - FJ3042088         ID -              ED -     1  OWN - HILFER
10:47:45.8257    21.3838   PDM    PD000 - SAVE     COMPLETE
10:47:45.8266    21.3838   USER   UT003 - EXIT CALLED BY  RAM2D1C
10:47:45.8290    21.3838   CSP    DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='CRJ3.MSG.'.
10:47:50.6917    21.3841   SCP     CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
10:47:50.6920    21.3841   SCP     CRAY TO VAX: FILE=$1$DUA107:[HILFER.FR2]CRJ3.MSG;1
10:47:50.6923    21.3841   SCP     CRAY TO VAX: 20 BYTES TRANSFERRED
10:47:53.8852    21.3845   USER   AUDIT.
10:48:10.3499    21.5929   USER   AU003 -       189 DATASETS,   208967 BLOCKS,   106927943 WORDS
10:48:10.3504    21.5930   USER   AU003 -        39 DATASETS,    29076 BLOCKS,    14876808 WORDS ONLINE
10:48:10.3509    21.5931   USER   AU003 -       150 DATASETS,   179891 BLOCKS,    92051135 WORDS OFFLINE
10:48:10.3589    21.5931   CSP    EXIT.
10:48:10.3605    21.5932   CSP    END OF JOB
10:48:10.3610    21.5932   CSP
10:48:10.3612    21.5932   CSP
10:48:10.5246    21.5933   USER         JOB NAME -                              CRJ3
10:48:10.5252    21.5933   USER         USER NUMBER -                           HILFER
10:48:10.5265    21.5933   USER         JOB SEQUENCE NUMBER -                       38889
```

CRJ3.CPR

```
10:48:10.5259     21.5933    USER
10:48:10.5265     21.5934    USER         TIME EXECUTING IN CPU -             0000:00:21.5933
10:48:10.5270     21.5934    USER         TIME WAITING TO EXECUTE -           0000:02:03.6841
10:48:10.5274     21.5934    USER         TIME WAITING FOR I O -              0000:00:39.5796
10:48:10.5278     21.5934    USER         TIME WAITING IN INPUT QUEUE         0000:00:00.0054
10:48:10.5282     21.5934    USER         MEMORY ' CPU TIME (MWDS'SEC) -             2.99854
10:48:10.5286     21.5935    USER         MEMORY ' I O WAIT TIME (MWDS'SEC) -        3.99978
10:48:10.5289     21.5935    USER         MINIMUM JOB SIZE (WORDS) -             43008
10:48:10.5293     21.5935    USER         MAXIMUM JOB SIZE (WORDS) -            215040
10:48:10.5297     21.5935    USER         MINIMUM FL (WORDS) -                   38400
10:48:10.5301     21.5935    USER         MAXIMUM FL (WORDS) -                  210432
10:48:10.5305     21.5935    USER         MINIMUM JTA (WORDS) -                   3584
10:48:10.5308     21.5935    USER         MAXIMUM JTA (WORDS) -                   5120
10:48:10.5312     21.5935    USER         DISK SECTORS MOVED -                    2952
10:48:10.5316     21.5935    USER         FSS SECTORS MOVED -                        0
10:48:10.5320     21.5935    USER         USER I O REQUESTS -                     1379
10:48:10.5323     21.5935    USER         USER I O SUSPENSIONS -                  1599
10:48:10.5327     21.5936    USER         OPEN CALLS -                              45
10:48:10.5331     21.5936    USER         CLOSE CALLS -                             44
10:48:10.5335     21.5936    USER         MEMORY RESIDENT DATASETS -                 0
10:48:10.5338     21.5936    USER         TEMPORARY DATASET SECTORS USED -           1
10:48:10.5342     21.5936    USER         PERMANENT DATASET SECTORS ACCESSED -    1414
10:48:10.5346     21.5936    USER         PERMANENT DATASET SECTORS SAVED -        346
10:48:10.5350     21.5936    USER         SECTORS RECEIVED FROM FRONT END -         15
10:48:10.5354     21.5936    USER         SECTORS QUEUED TO FRONT END -              1
10:48:10.8906     21.6012    USER
10:48:10.8909     21.6012    USER    ..........................................................................
10:48:10.8913     21.6012    USER                        '''   COST TABLE FOR THIS JOB '''
10:48:10.8916     21.6013    USER                   JOBNAME -----------                         CRJ3
10:48:10.8920     21.6014    USER                   USER IDENT ---------                        HILFER
10:48:10.8924     21.6015    USER                   BEGAN EXECUTION ----  WED APR 20, 1988      10:45:04  HOURS.
10:48:10.8928     21.6016    USER                   AT A PRIORITY OF  --                              3
10:48:10.8931     21.6017    USER                   AND JOB CLASS OF  --                        DSMALL
10:48:10.8935     21.6018    USER         21.599948 SECONDS OF CPU TIME       @ $ 630.00  HR    --   $       3.78
10:48:10.8939     21.6020    USER          2.998950 MEMORY'CPU (MWRD-SEC)     @ $  84.00  HR    --   $       0.07
10:48:10.8943     21.6021    USER          4.004100 MEMORY'I O (MWRD-SEC)     @ $  84.00  HR    --   $       0.09
10:48:10.8947     21.6022    USER          0.002954 I O MEGASECTORS MOVED     @ $  84.00  EA    --   $       0.25
10:48:10.8951     21.6023    USER          0.000000 TAPE MOUNT(S)             @ $   5.00  EA    --   $       0.00
10:48:10.8955     21.6024    USER
10:48:10.8958     21.6024    USER                   ''''    TOTAL COST FOR THIS JOB    ''''   --   $       4.19
10:48:10.8961     21.6024    USER    ..........................................................................
```

CPJ3.CPR

```
10:52:01 9667    0 0000   CSP    ......................................................................
10:52:01 9670    0 0000   CSP    .                                                                    .
10:52:01 9673    0 0000   CSP    .                     WELCOME TO THE NRL CRAY XMP                    .
10:52:01 9676    0 0000   CSP    .                                                                    .
10:52:01 9679    0 0001   CSP    ......................................................................
10:52:01 9682    0.0001   CSP    .  There will be no CRAY off-line data set recalls on Tuesday or Wednesdav .
10:52:01 9685    0.0001   CSP    .  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP .
10:52:01 9688    0.0001   CSP    .  runs on our CRAY archive tape library.                             .
10:52:01 9691    0 0001   CSP    ......................................................................
10:52:01 9715    0 0001   CSP           CRAY X-MP SERIAL 415 65    NAVAL RESEARCH LABORATORY    04 20 88
10:52:01 9718    0 0001   CSP
10:52:01.9722    0.0001   CSP           CRAY OPERATING SYSTEM          COS 1.15  ASSEMBLY DATE 01 04 88
10:52:01 9725    0.0001   CSP
10:52:01.9728    0 0002   CSP
10:52:01 9949    0 0002   CSP     JOB.JN-CPJ3.MFL-511000.US-DEFER.
10:52:02 0243    0 0014   CSP     ACCOUNT.AC-.US-.UPW-.APW-
10:52:03 4267    0 1114   USER   AC213 - '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
10:52:04 0746    0 1146   USER   AUDIT.
10:52:31 6486    0 3240   USER   AU003 -       189 DATASETS,   208967 BLOCKS,    106927943 WORDS
10:52:31 6491    0.3241   USER   AU003 -        39 DATASETS,    29076 BLOCKS,     14876808 WORDS ONLINE
10:52:31 6496    0 3242   USER   AU003 -       150 DATASETS.   179891 BLOCKS,     92051135 WORDS OFFLINE
10:52:31 6581    0 3246   CSP    ACCESS.  DN-DISLIB.ID-DISSPLA.OWN-LIBRARY.
10:52:31 9239    0 3246   PDM    PD000 - PDN - DISLIB            ID - DISSPLA   ED -    1  OWN - LIBRARY
10:52:31 9242    0 3246   PDM    PD000 - ACCESS  COMPLETE
10:52:31 9261    0 3250   CSP    ACCESS.  DN-INTLIB.ID-DISSPLA.OWN-LIBRARY.
10:52:32.1634    0.3250   PDM    PD000 - PDN - INTLIB            ID - DISSPLA   ED -    1  OWN - LIBRARY
10:52:32 1638    0 3250   PDM    PD000 - ACCESS  COMPLETE
10:52:32 1862    0.3253   CSP    ACCESS.  DN-DVSD.ID-DISSPLA.OWN-LIBRARY.
10:52:32 4003    0 3254   PDM    PD000 - PDN - DVSD              ID - DISSPLA   ED -    1  OWN - LIBRARY
10:52:32 4006    0 3254   PDM    PD000 - ACCESS  COMPLETE
10:52:32.4022    0 3254   CSP    FETCH.   DN-PJ3.TEXT-'PRAM1CD.FOR'.
10:52.39 7022    0 3256   SCP     VAX TO CRAY: %SYSTEM-S-NORMAL, normal successful completion
10 52:39 7025    0 3256   SCP     VAX TO CRAY: FILE-$1$DUA107:[HILFER FR2]PRAM1CD.FOR;9
10 52:39 7028    0 3256   SCP     VAX TO CRAY: 177688 BYTES TRANSFERRED
10:52:42 1554    0 3256   SCP    SS004 - DATASET RECEIVED FROM FRONT END
10:52:42 5372    0 3260   CSP    CFT.      I-PJ3.ON-INZ.
10:52:43.0746    0.3263   USER   CF000 - CFT VERSION -   06 16 87 1.15BF2
10:53:01 3381    8 8122   USER   CF001 - COMPILE TIME -     8.4859 SECONDS
10:53:01 3428    8 8122   USER   CF002 -       3958 LINES,      2705 STATEMENTS
10:53:01 3432    8 8122   USER   CF003 -    107850 WORDS.    14540 I/O BUFFERS USED
10:53:01 4860    8 8128   CSP    LDR.      LIB-INTLIB:DISLIB.NX.AB-XPJ3.
10:53:08 0174   10.1233   CSP    SAVE.     DN-XPJ3.
10:53:08.2797   10.1233   PDM    PD000 - PDN - XPJ3              ID -           ED -    8  OWN - HILFER
10:53:08 2800   10.1233   PDM    PD000 - SAVE    COMPLETE
10 53:08 2819   10 1234   CSP    RELEASE. DN-PJ3.
10.53:08 2858   10 1235   CSP    FETCH.    DN-NPRAM1.TEXT-'NPJ3 DAT'.
10:53:13 5783   10 1236   SCP     VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
10.53:13.5786   10.1236   SCP     VAX TO CRAY: FILE-$1$DUA107:[HILFER.FR2]NPJ3.DAT:9
10:53:13 5789   10.1236   SCP     VAX TO CRAY: 1056 BYTES TRANSFERRED
10:53.17 7079   10.1236   SCP    SS004 - DATASET RECEIVED FROM FRONT END
10:53:17 9790   10.1238   CSP    XPJ3.
10 53:20 9282   10 1284   PDM    PD000 - PDN - FJ3042088         ID -           ED -    1  OWN - HILFER
10:53:20 9286   10 1284   PDM    PD000 - ACCESS  COMPLETE
10.54.05 5294   24.0181   USER   UT003 - EXIT CALLED BY  PRAM1CD
10:54:05 5320   24 0181   CSP    DISPOSE. DN-META.DF-BB.WAIT.TEXT-'PLT2.DAT'.
10.54:21 1493   24.0183   SCP     CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
10:54:21.1496   24.0183   SCP     CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]PLT2.DAT;1
10:54:21 1500   24 0183   SCP     CRAY TO VAX: 547920 BYTES TRANSFERRED
10:54:23 8585   24 0184   CSP    DISPOSE. DN-EPRM.DF-BB.WAIT.TEXT-'CPJ3.MSG.'.
10:54:29 7490   24 0186   SCP     CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
```

CPJ3.CPR

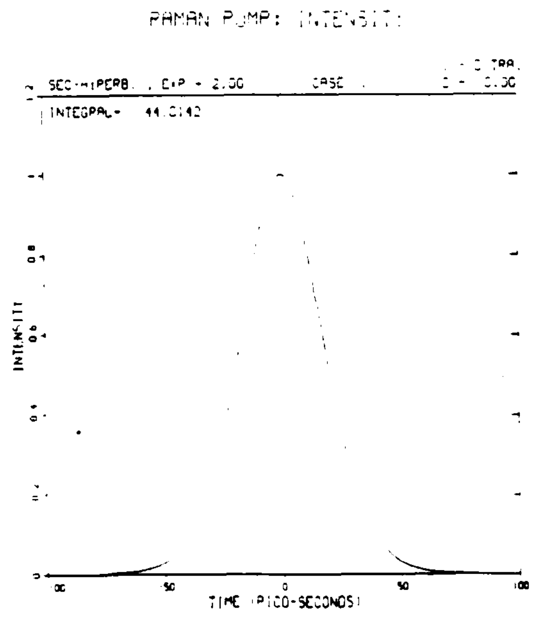
```
10:54:29.7493     24.0188     SCP     CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]CPJ3.MSG;1
10:54:29.7496     24.0188     SCP     CRAY TO VAX: 3404 BYTES TRANSFERRED
10:54:34.6297     24.0186     CSP    DISPOSE. DN-META.DF-BB.WAIT.TEXT- CPJ3.DSP .
10:54:34.6315     24.0188     EXP    SY001 - RLS COULD NOT FIND A DNT FOR    META
10:54:35.1661     24.0192     USER   AUDIT.
10:55:05.6981     24.2272     USER   AU003          190 DATASETS.   209325 BLOCKS.   107111217 WORDS
10:55:05.6986     24.2272     USER   AU003 -         40 DATASETS.    29434 BLOCKS.    15060082 WORDS ONLINE
10:55:05.6990     24.2273     USER   AU003 -        150 DATASETS.   179891 BLOCKS.    92051135 WORDS OFFLINE
10:55:05.8647     24.2274     CSP    EXIT.
10:55:05.8666     24.2274     CSP    END OF JOB
10:55:05.8669     24.2274     CSP
10:55:05.8674     24.2274     CSP
10:55:06.0309     24.2276     USER       JOB NAME -                                  CPJ3
10:55:06.0312     24.2276     USER       USER NUMBER -                               HILFER
10:55:06.0316     24.2276     USER       JOB SEQUENCE NUMBER -                           38928
10:55:06.0319     24.2276     USER
10:55:06.0323     24.2276     USER       TIME EXECUTING IN CPU -                     0000:00:24.2275
10:55:06.0327     24.2276     USER       TIME WAITING TO EXECUTE -                   0000:02:01.7699
10:55:06.0330     24.2276     USER       TIME WAITING FOR I/O -                      0000:00:36.4403
10:55:06.0334     24.2276     USER       TIME WAITING IN INPUT QUEUE -               0000:00:00.0166
10:55:06.0338     24.2277     USER       MEMORY * CPU TIME (MWDS*SEC) -                    4.65757
10:55:06.0342     24.2277     USER       MEMORY * I/O WAIT TIME (MWDS*SEC) -               4.39400
10:55:06.0345     24.2277     USER       MINIMUM JOB SIZE (WORDS) -                    43008
10:55:06.0349     24.2277     USER       MAXIMUM JOB SIZE (WORDS) -                   311296
10:55:06.0651     24.2277     USER       MINIMUM FL (WORDS) -                          38400
10:55:06.0654     24.2277     USER       MAXIMUM FL (WORDS) -                         306176
10:55:06.0658     24.2277     USER       MINIMUM JTA (WORDS) -                          3584
10:55:06.0661     24.2277     USER       MAXIMUM JTA (WORDS) -                          5120
10:55:06.0665     24.2278     USER       DISK SECTORS MOVED -                           4492
10:55:06.0669     24.2278     USER       FSS SECTORS MOVED -                               0
10:55:06.0672     24.2278     USER       USER I/O REQUESTS -                            1553
10:55:06.0676     24.2278     USER       USER I/O SUSPENSIONS -                         1877
10:55:06.0679     24.2278     USER       OPEN CALLS -                                     51
10:55:06.0683     24.2278     USER       CLOSE CALLS -                                    49
10:55:06.0686     24.2278     USER       MEMORY RESIDENT DATASETS -                        0
10:55:06.0690     24.2278     USER       TEMPORARY DATASET SECTORS USED -              183
10:55:06.0694     24.2278     USER       PERMANENT DATASET SECTORS ACCESSED -         2451
10:55:06.0697     24.2278     USER       PERMANENT DATASET SECTORS SAVED -             358
10:55:06.0701     24.2278     USER       SECTORS RECEIVED FROM FRONT END -              45
10:55:06.0704     24.2278     USER       SECTORS QUEUED TO FRONT END -                 138
10:55:06.3479     24.2355     USER
10:55:06.3481     24.2355     USER   ..........................................................................................
10:55:06.3485     24.2356     USER                          ***   COST TABLE FOR THIS JOB ***
10:55:06.3489     24.2357     USER                       JOBNAME -----------                         CPJ3
10:55:06.3493     24.2358     USER                       USER IDENT.---------                        HILFER
10:55:06.3497     24.2359     USER                       BEGAN EXECUTION ----  WED APR 20, 1988     10:52:01  HOURS
10:55:06.3500     24.2360     USER                       AT A PRIORITY OF  --                               3
10:55:06.3504     24.2361     USER                       AND JOB CLASS OF  --                        DSMALL
10:55:06.3508     24.2362     USER          24.234252  SECONDS OF CPU TIME       @ $ 630.00  HR     --    $          4.24
10:55:06.3512     24.2363     USER           4.658011  MEMORY*CPU (MWRD-SEC)     @ $  84.00  HR     --    $          0.11
10:55:06.3517     24.2364     USER           4.397854  MEMORY*I/O (MWRD-SEC)     @ $  84.00  HR     --    $          0.10
10:55:06.3521     24.2366     USER           0.004494  I/O MEGASECTORS MOVED     @ $  84.00  EA     --    $          0.38
10:55:06.3525     24.2367     USER           0.000000  TAPE MOUNT(S)             @ $   5.00  EA     --    $          0.00
10:55:06.3529     24.2368     USER
10:55:06.3531     24.2368     USER                       ****    TOTAL COST FOR THIS JOB     ****    --    $          4.83
10:55:06.3534     24.2368     USER   ..........................................................................................
```

PLT2.DAT (Example A2)

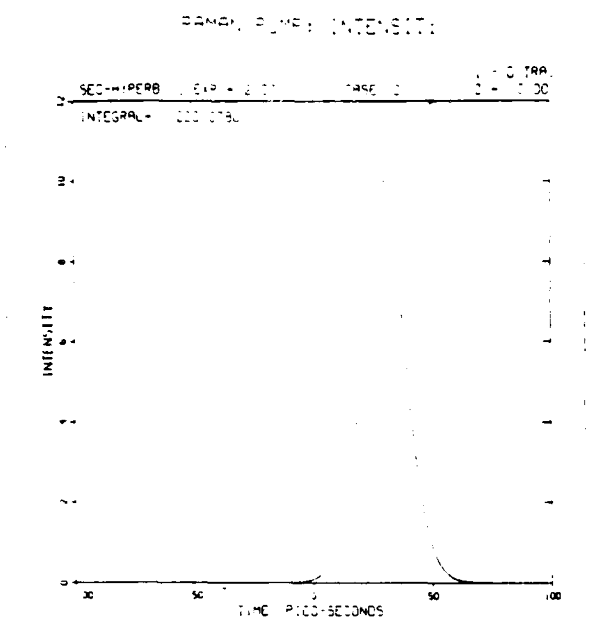
```
LIST OF INPUT PARAMETERS

ICOND    -      1
NMAX     -      4000
NPUMP    -      2
NT       -      1024
NT       -      1
GAIN     -       0.0000
PHST     -       0.0000
RALASH   -       5.0000
RAMASH   -        .5000
RIST     -        [illegible]
RKP      -       [illegible]
RKS      -       [illegible]
TOC      -       5.0000
TOST     -      -40.000
TTWO     -       633.00
TWST     -       40.000
ZFINAL   -       100.00
ZKEEP    -       50.000
ZSTEP    -       0.0500
```

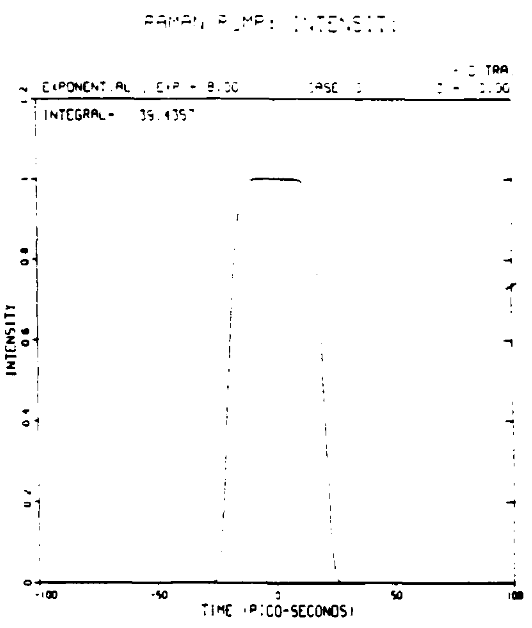
```
LIST OF INPUT PARAMETERS  (CONTD)

ITYPE        -       1          4          1          1          1
PHL(1-10)    -     0.0000     0.0000     6.2800     0.0000     0.0000
                   0.0000     0.0000     0.0000     0.0000     0.0000
RINT(1-10)   -     1.0000    10.0000     1.0000     0.5500     0.5500
                   0.5500     0.5500     0.5500     0.5500     0.5500
RTYPE        -     2.0000     2.0000     8.0000     2.0000     2.0000
                   2.0000     2.0000     2.0000
TM(1,2)      -    -100.00     100.00
TOFF(1-10)   -     0.0000     30.000     0.0000     0.0000     0.0000
                   0.0000     0.0000     0.0000     0.0000     0.0000
TWIDTH       -     40.000     20.000     40.000     40.000     40.000
                   40.000     40.000     40.000     40.000     40.000
```

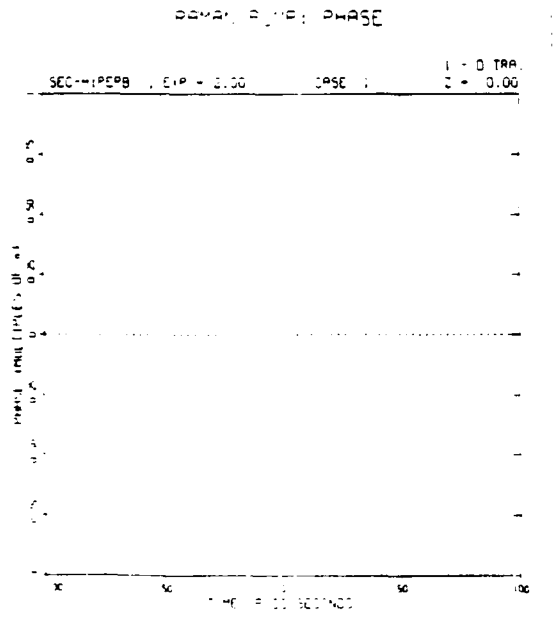
```
LIST OF INPUT PARAMETERS  (CONTD)

ISEC(1-9,1-9)  -
    1.000      2.000      2.000      2.000      3.000      2.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    1.000      2.000      2.000      2.000      3.000      2.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    1.000      2.000      2.000      2.000      3.000      2.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    1.000      2.000      2.000      2.000      3.000      2.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000

  [illegible]  0.000      0.000      0.000      0.000    [illegible]  0.000      0.000
  [illegible]  0.000      0.000      0.000      0.000    [illegible]  0.000      0.000

  [illegible]  0.000      0.000      0.000      0.000    [illegible]  0.000      0.000
  [illegible]  0.000      0.000      0.000    [illegible] [illegible]  0.000      0.000

  [illegible]  2.000      2.000      2.000      3.000    [illegible]  0.000      0.000
  [illegible]  0.000      0.000      0.000    [illegible] [illegible]  0.000      0.000

  [illegible] [illegible]  2.000      2.000    [illegible] [illegible]  0.000      0.000
  [illegible] [illegible]  0.000      0.000    [illegible] [illegible]  0.000      0.000

  [illegible] [illegible]  0.000    [illegible] [illegible] [illegible]  0.000      0.000
  [illegible]  0.000      0.000      0.000    [illegible] [illegible]  0.000      0.000

  [illegible] [illegible]  0.000    [illegible] [illegible] [illegible]  0.000      0.000
    0.000      0.000      0.000    [illegible] [illegible] [illegible]  0.000      0.000

    0.000      0.000      0.000      0.000    [illegible] [illegible]  0.000      0.000
    0.000      0.000      0.000      0.000    [illegible] [illegible]  0.000    [illegible]

    0.000      0.000      0.000      0.000    [illegible] [illegible]  0.000      0.000
    0.000      0.000      0.000      0.000      0.000    [illegible]  0.000      0.000

    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
    0.000      0.000      0.000      0.000      0.000      0.000      0.000      0.000
```

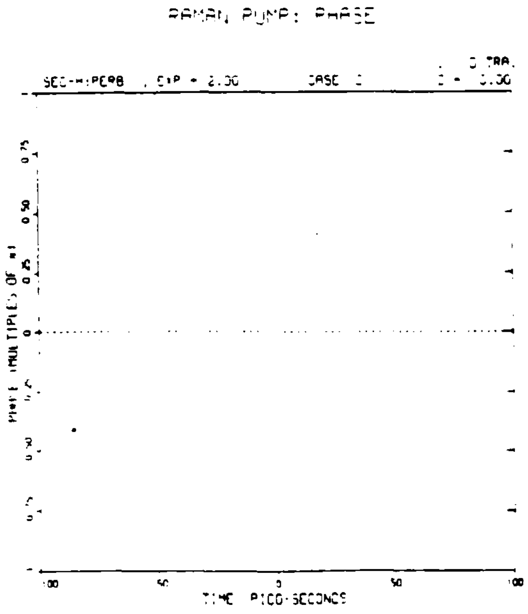


PLT2.DAT (Example A2)

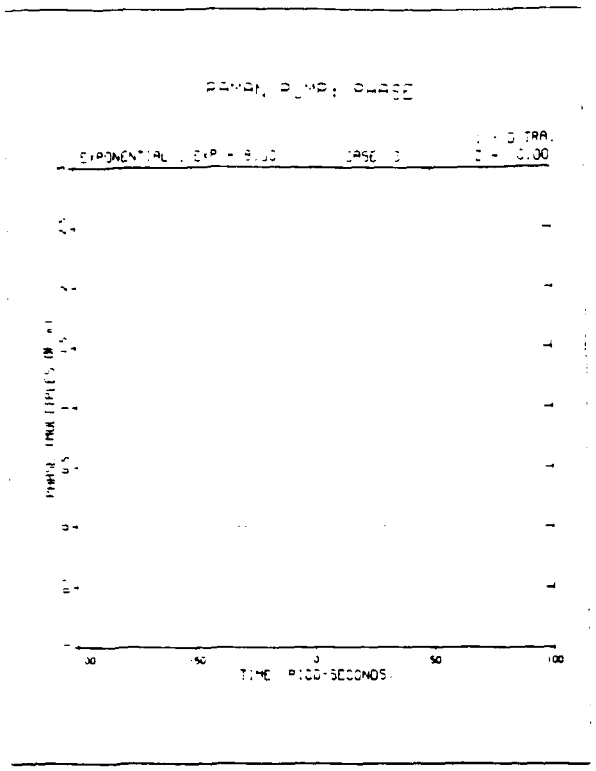


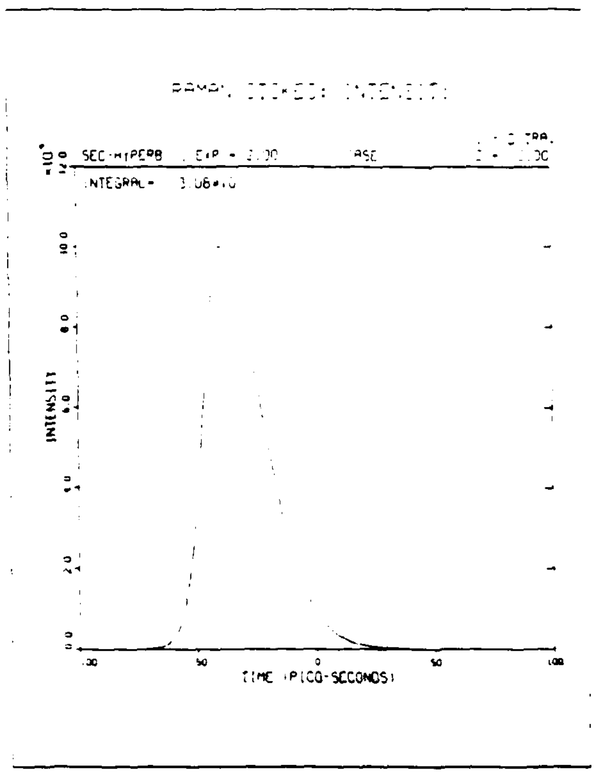


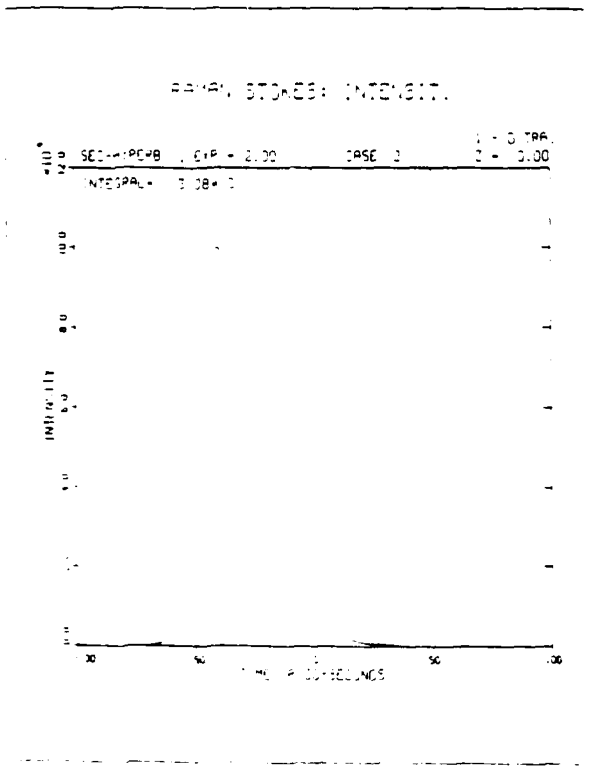


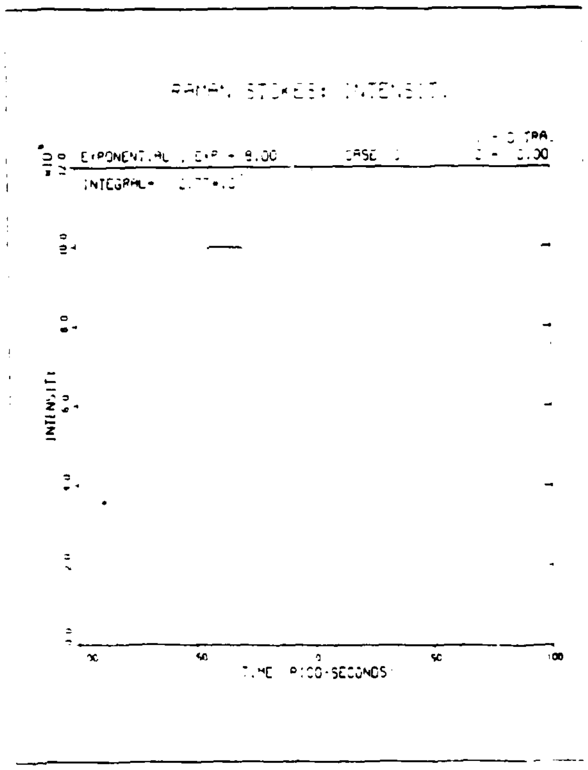


PLT2.DAT (Example A2)

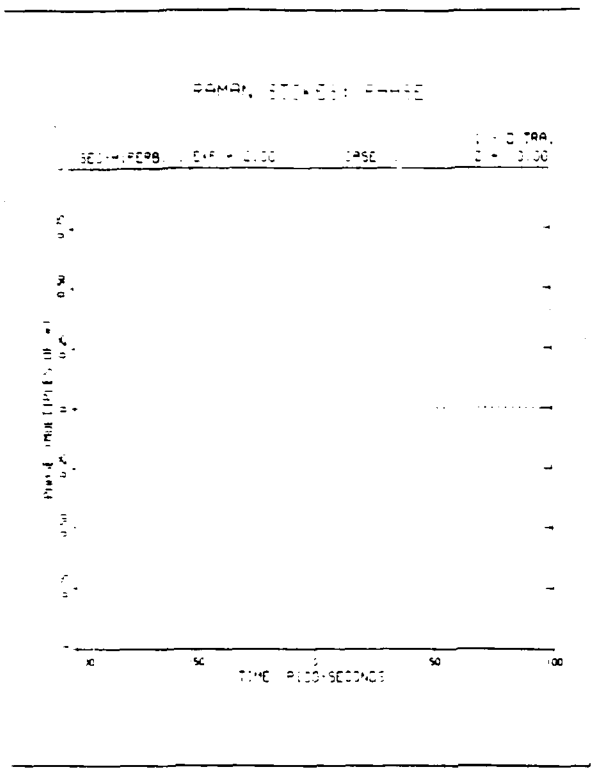


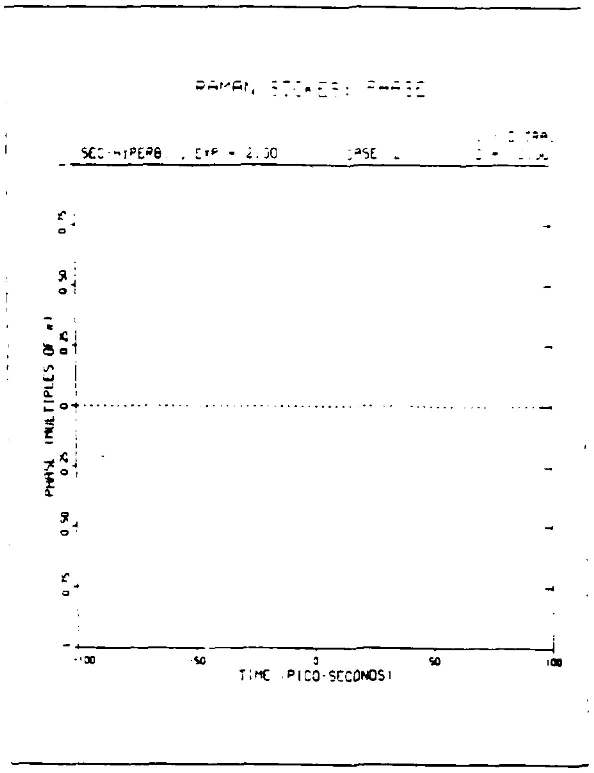


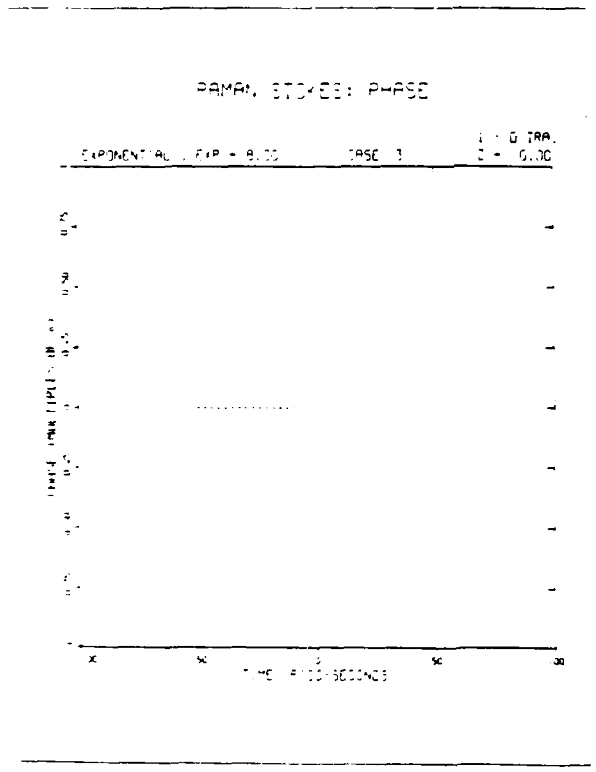


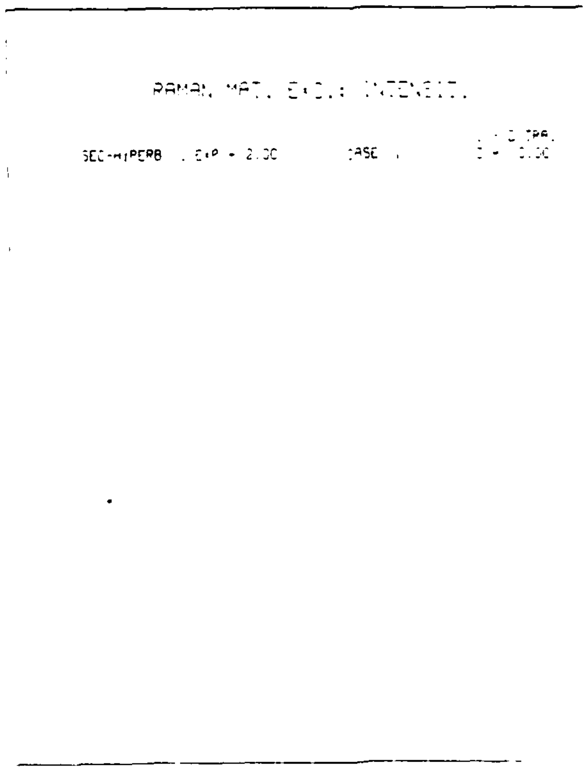


PLT2.DAT (Example A2)

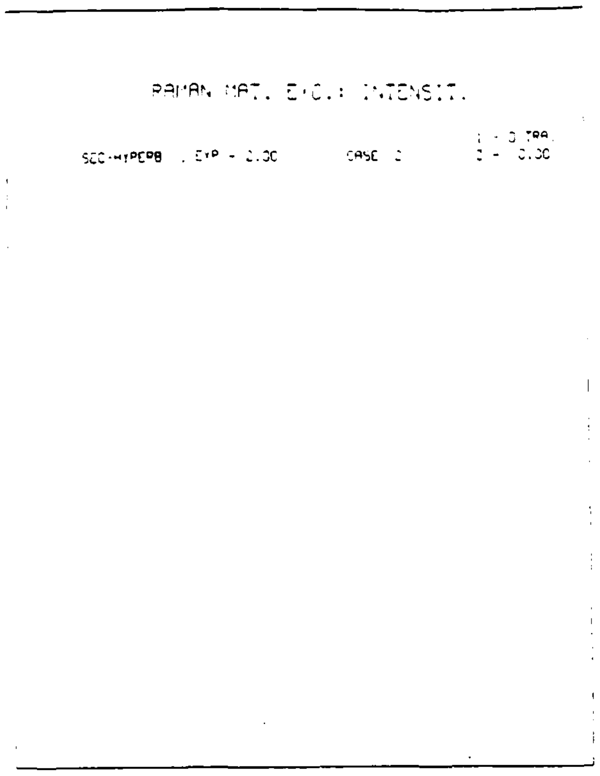


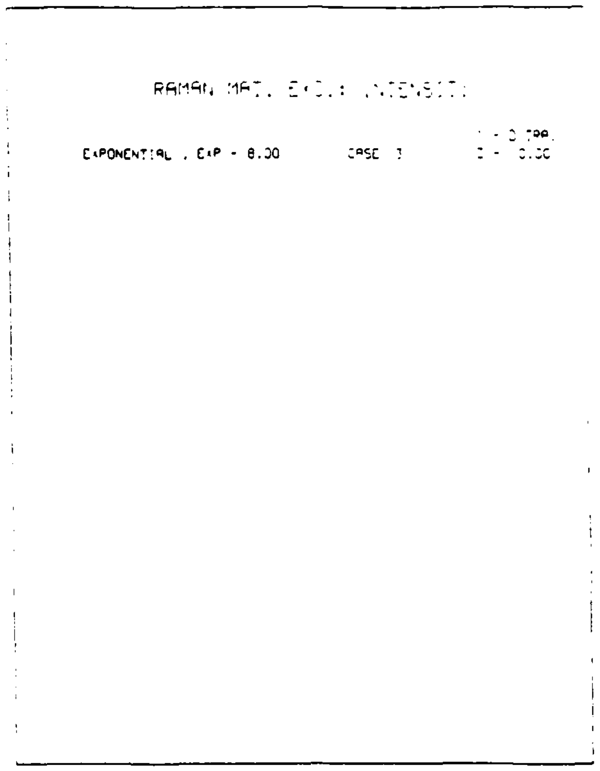


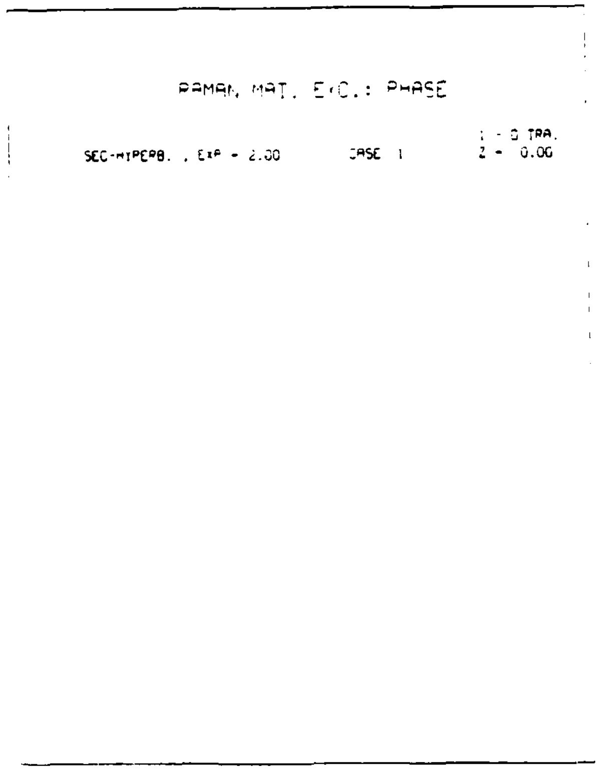


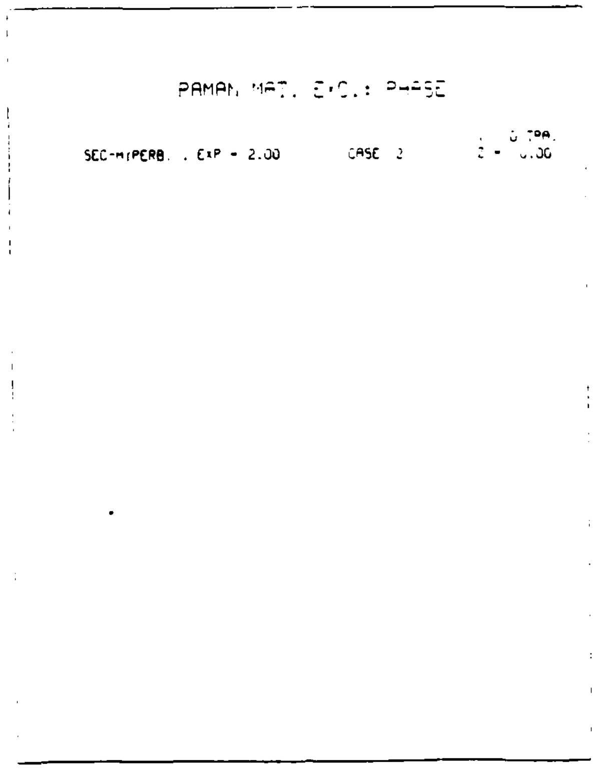


PLT2.DAT (Example A2)

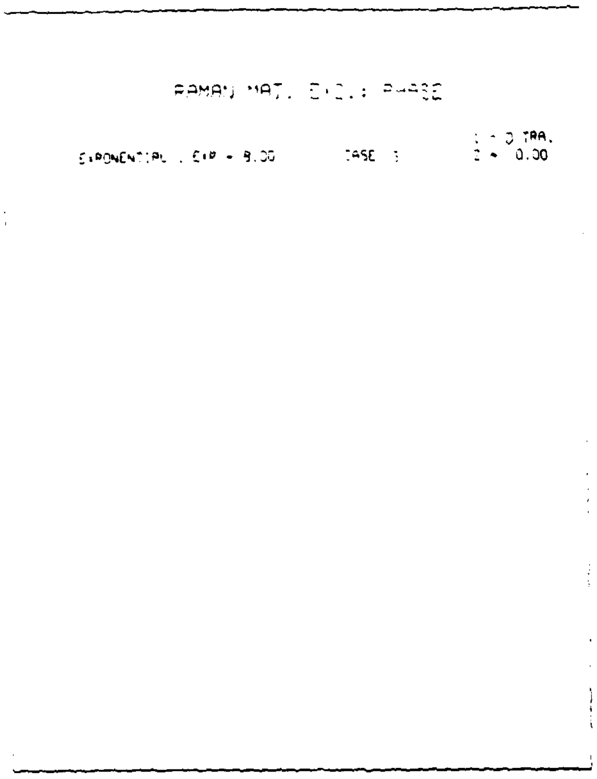


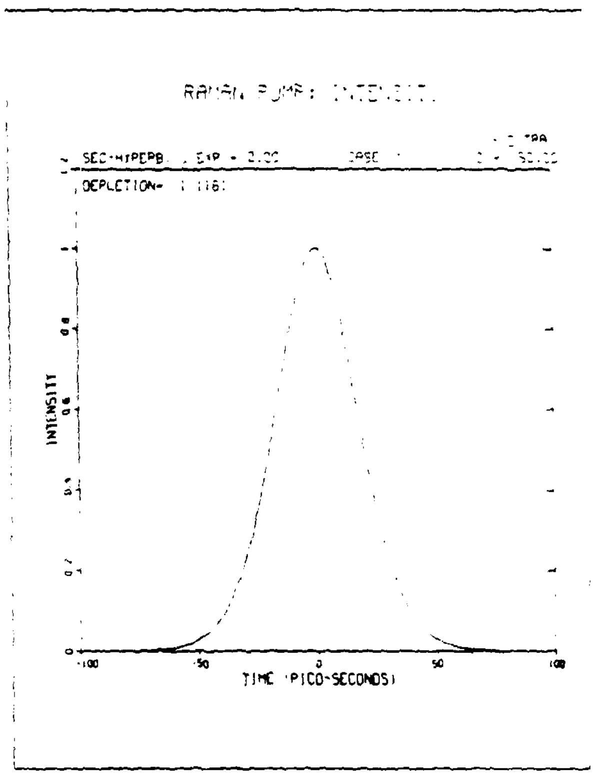


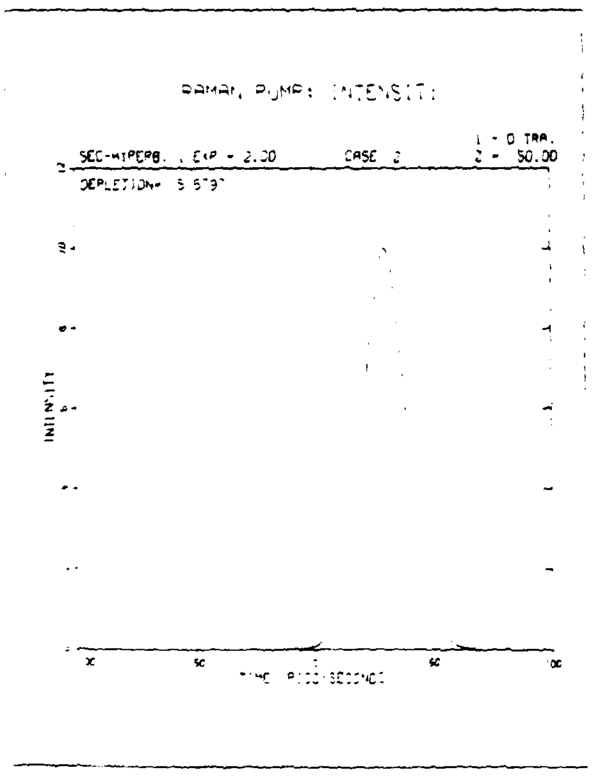


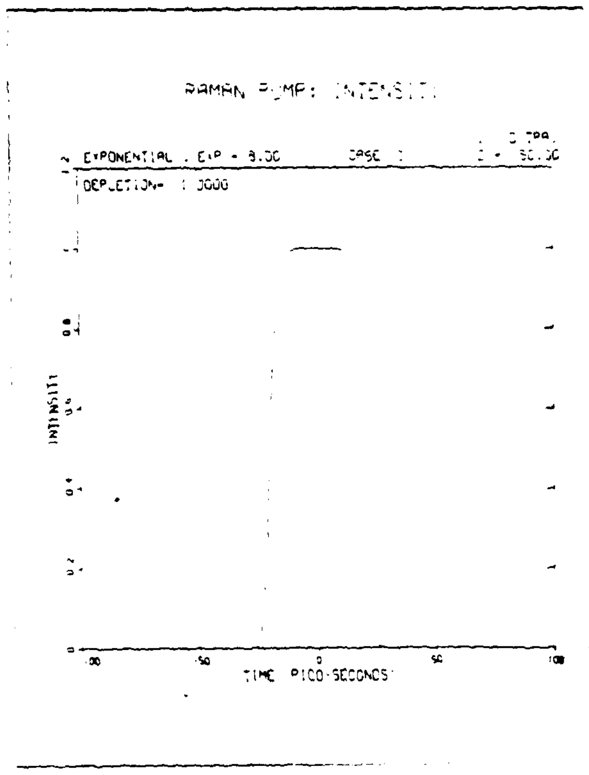


PLT2.DAT (Example A2)

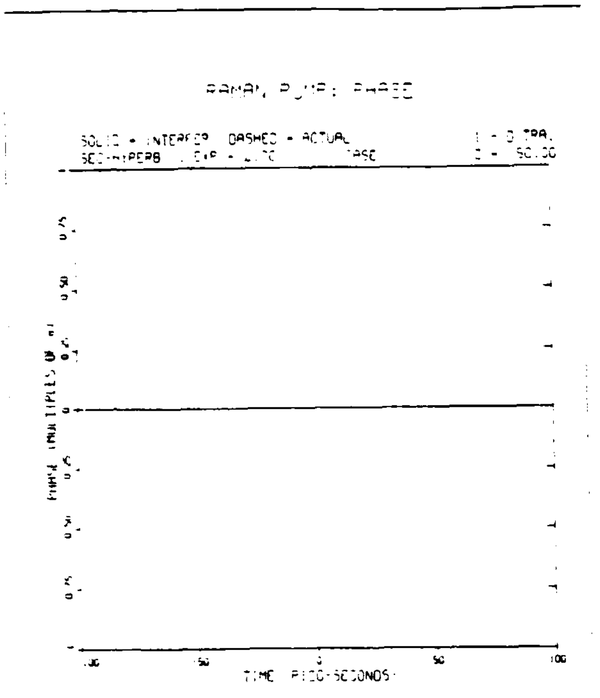


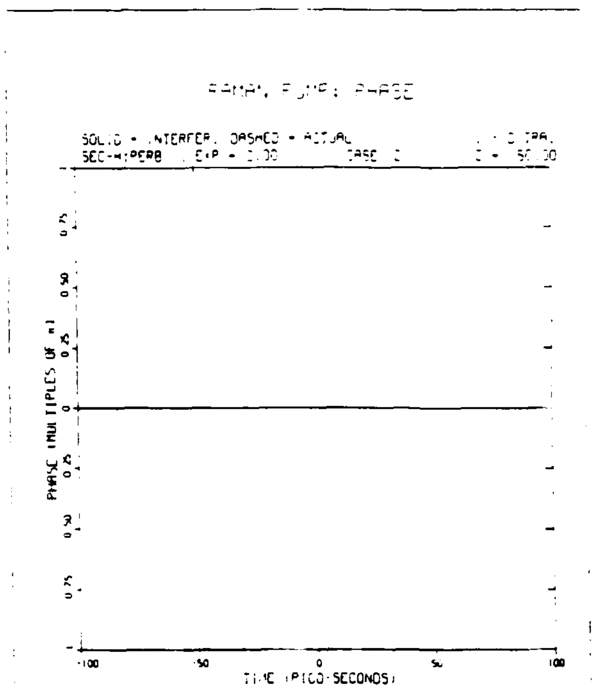


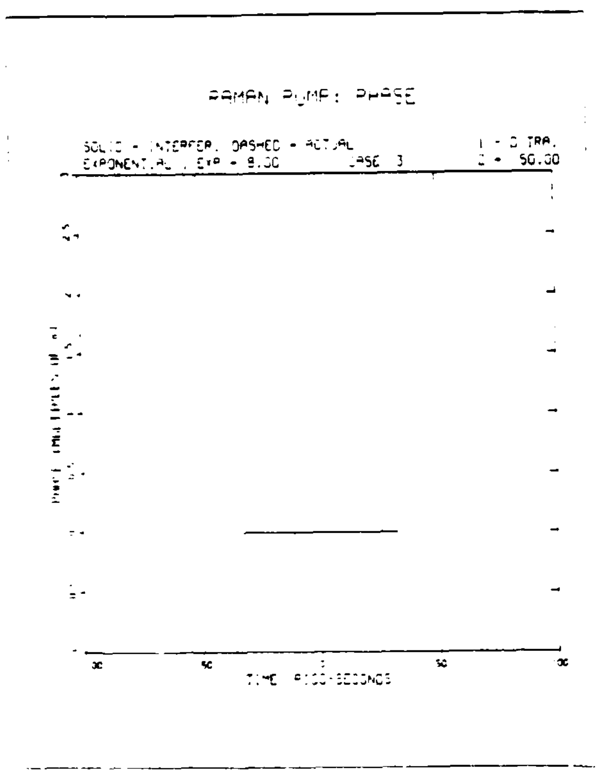


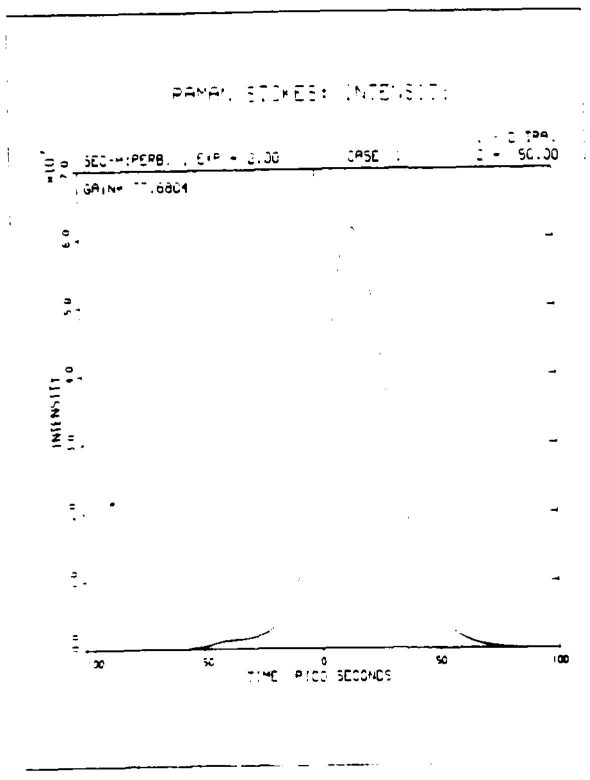


PLT2.DAT (Example A2)

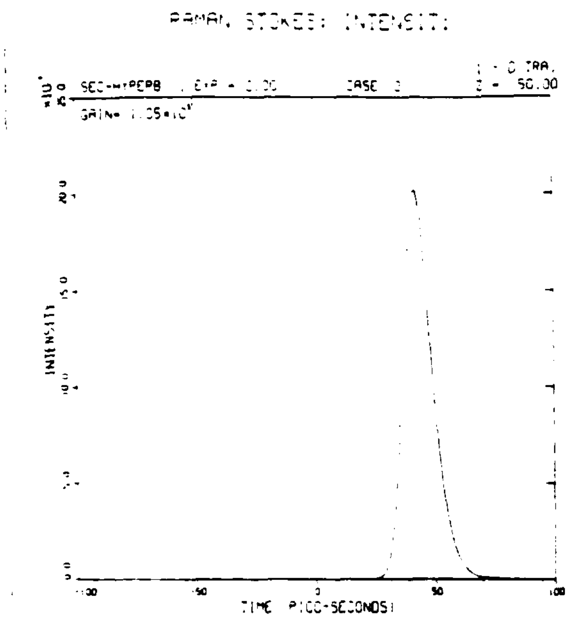


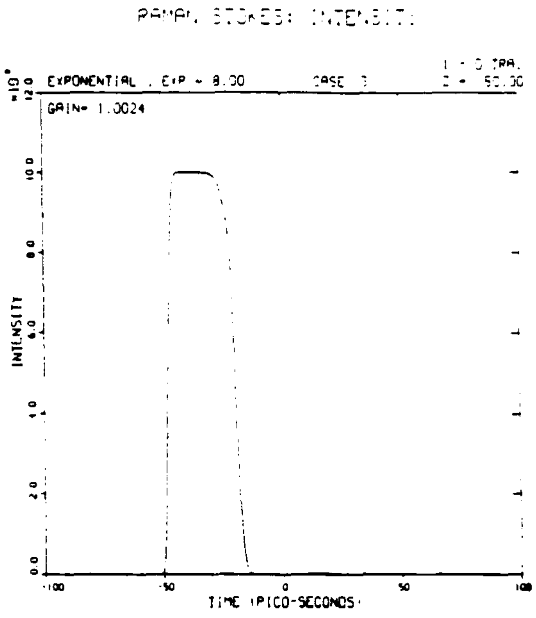


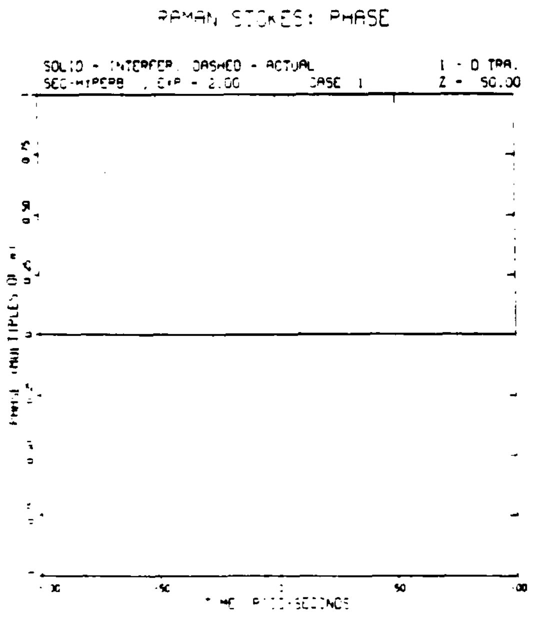


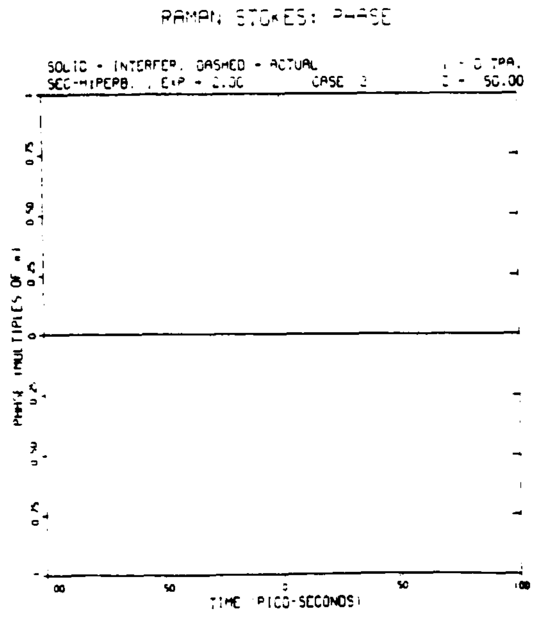


PLT2.DAT (Example A2)

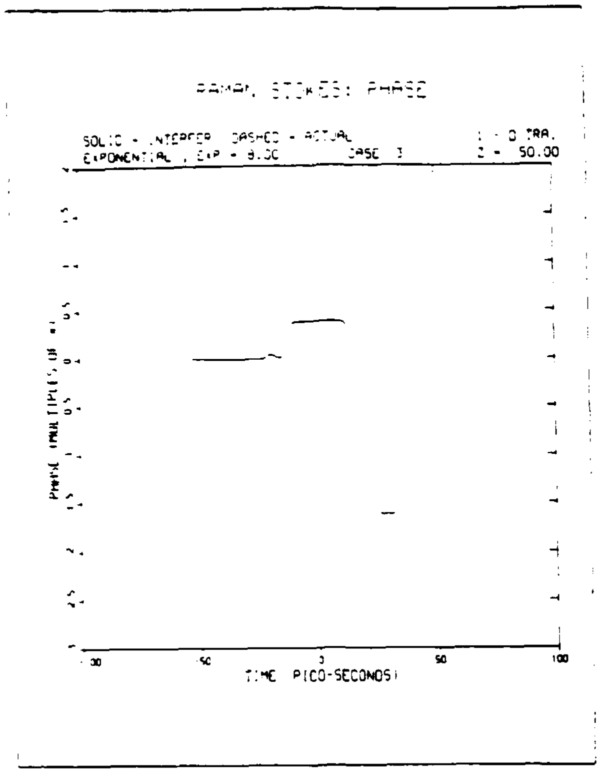


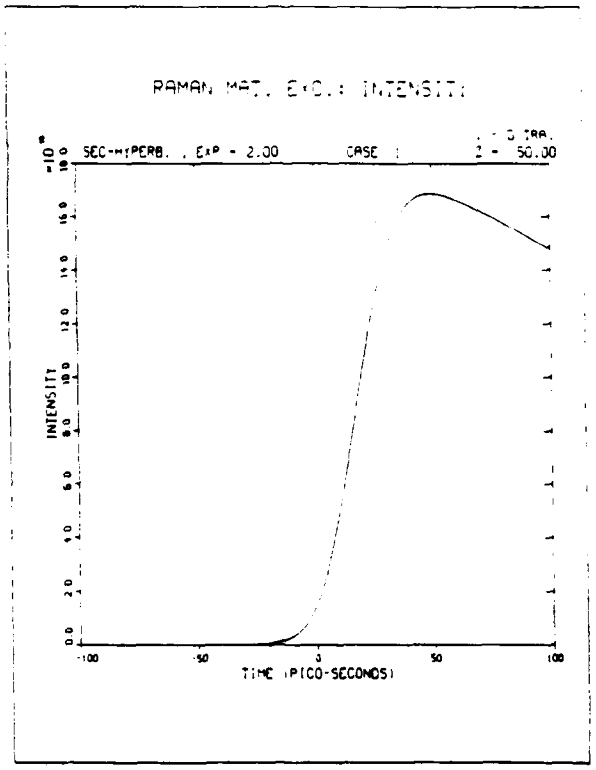


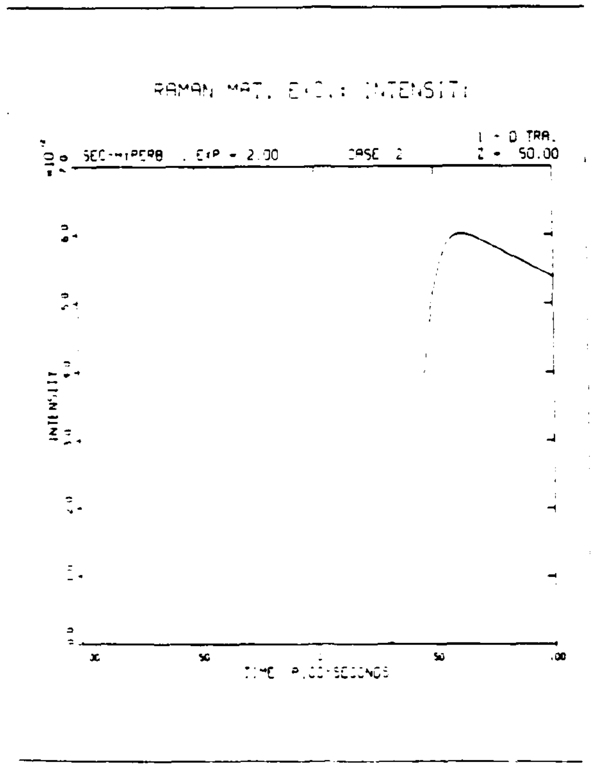


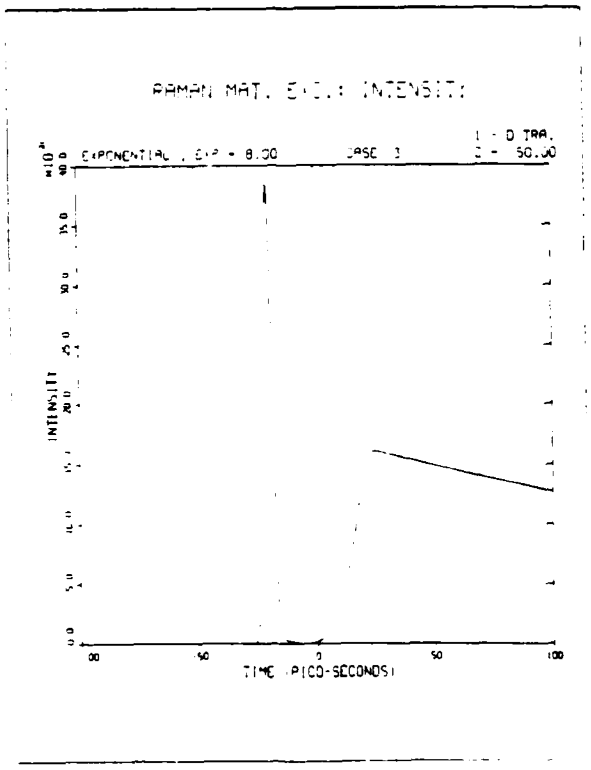


PLT2.DAT (Example A2)

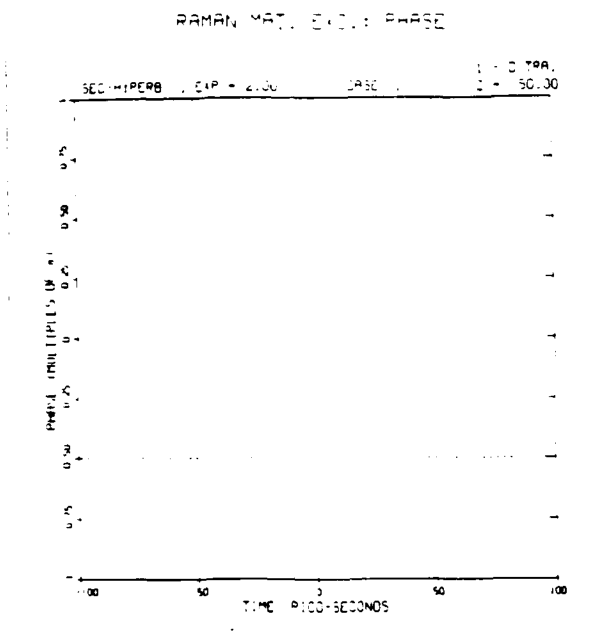


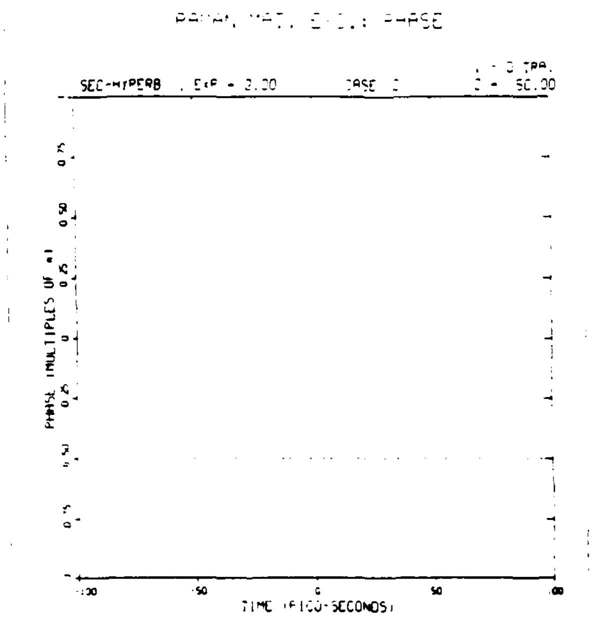


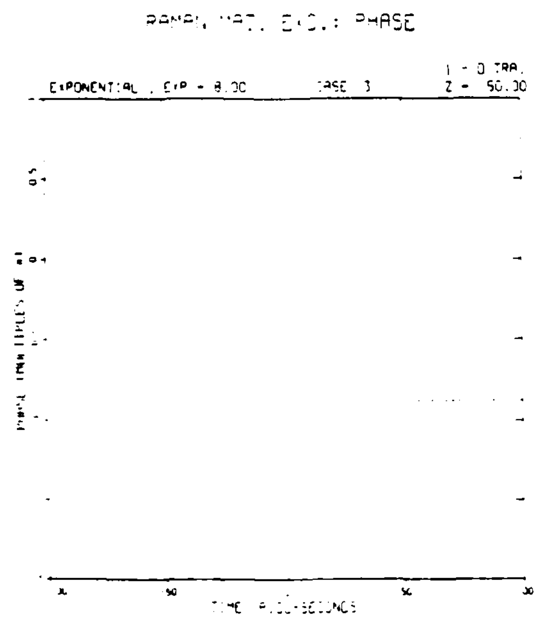


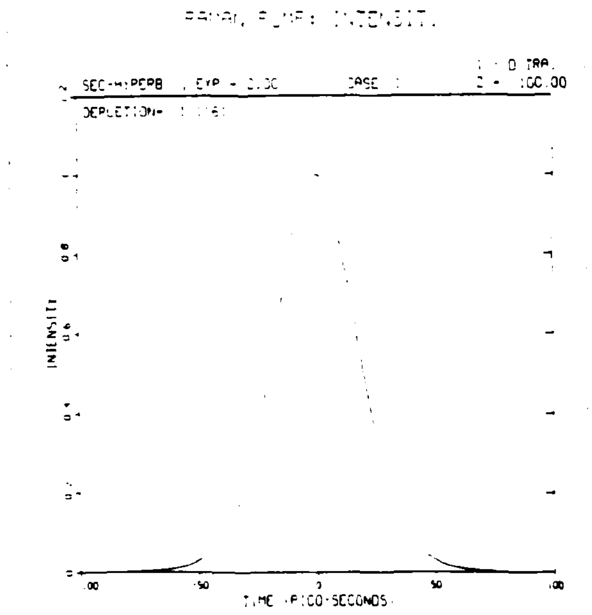


PLT2.DAT (Example A2)

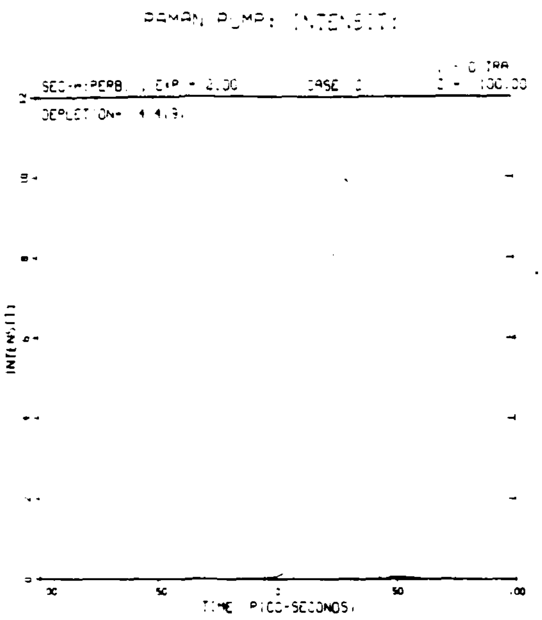


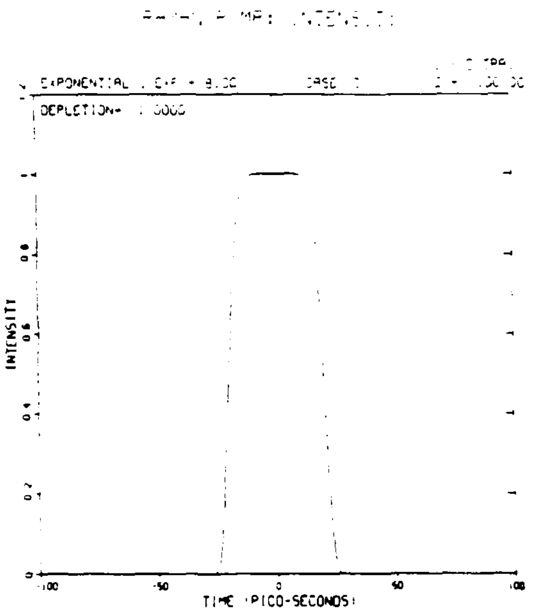


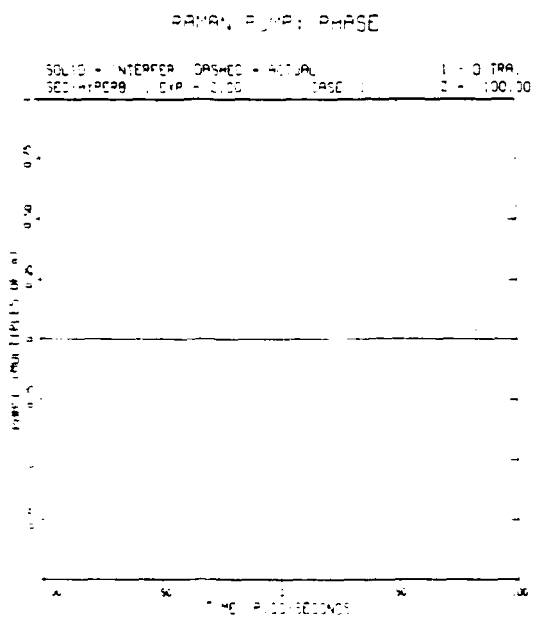


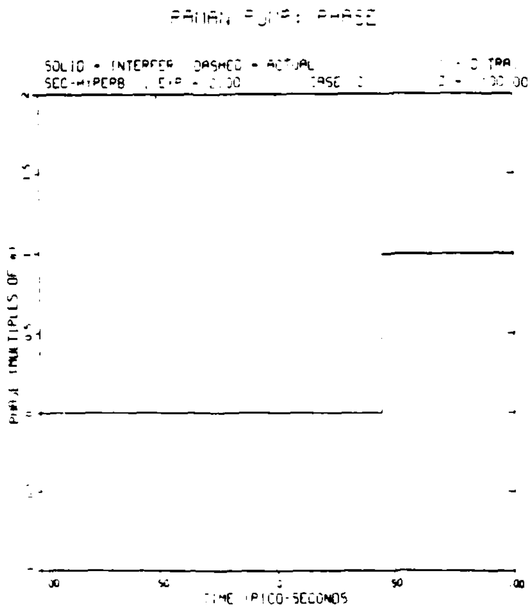


PLT2.DAT (Example A2)

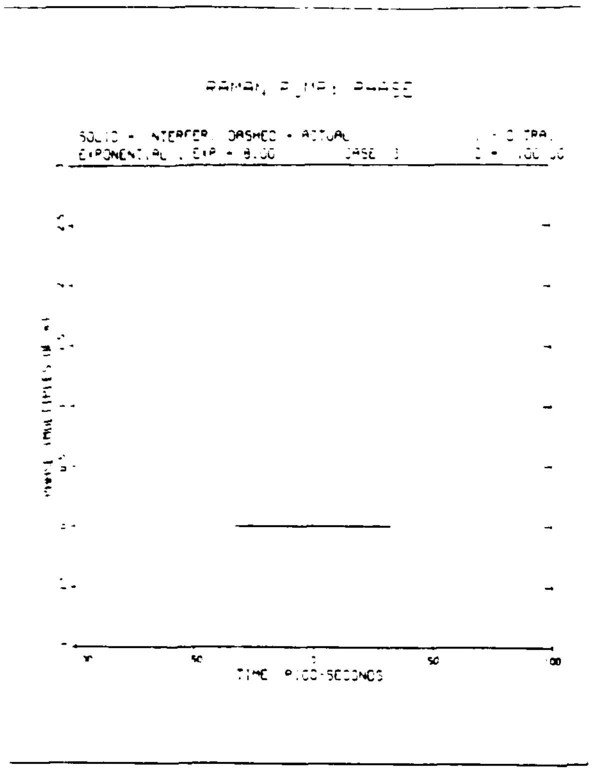


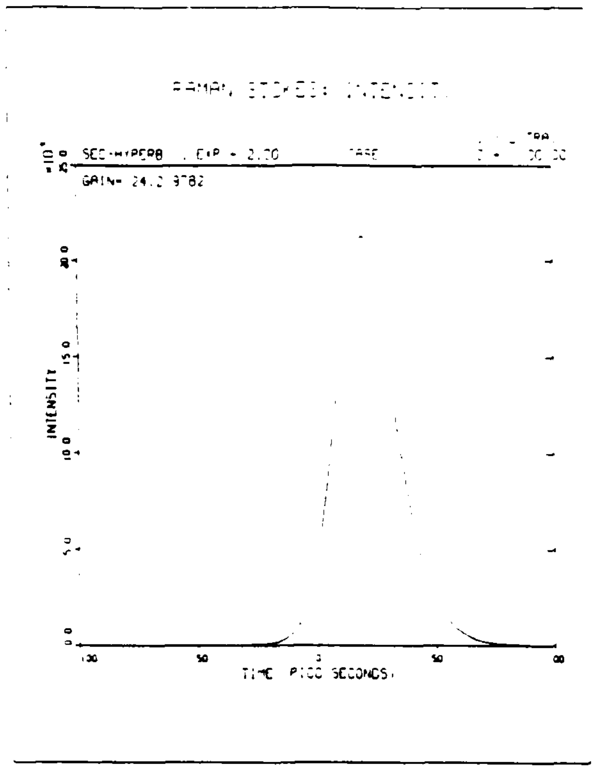


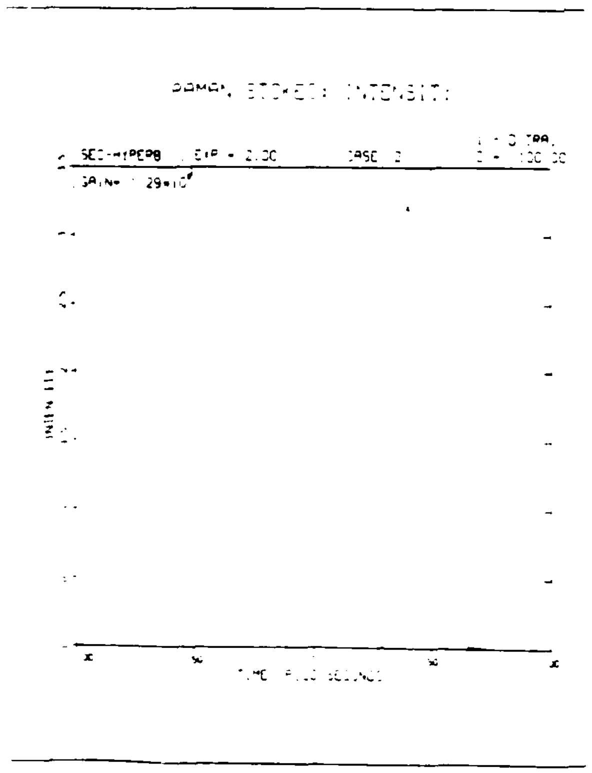


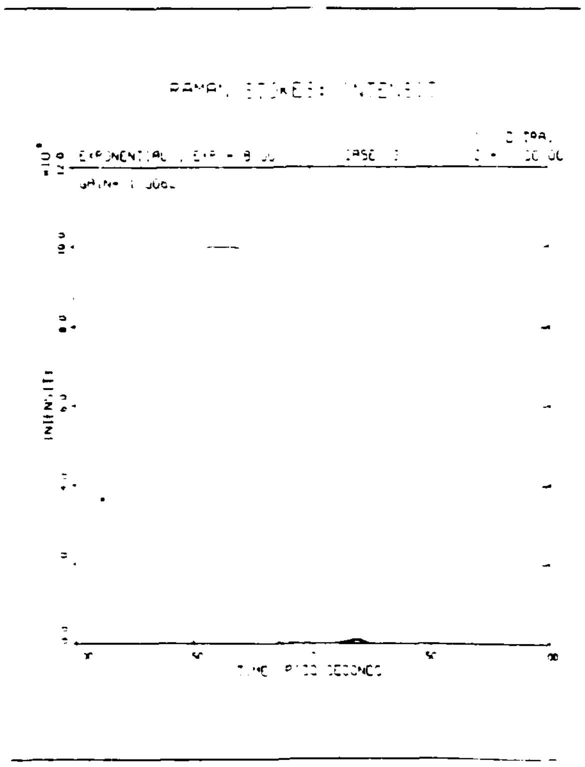


PLT2.DAT (Example A2)

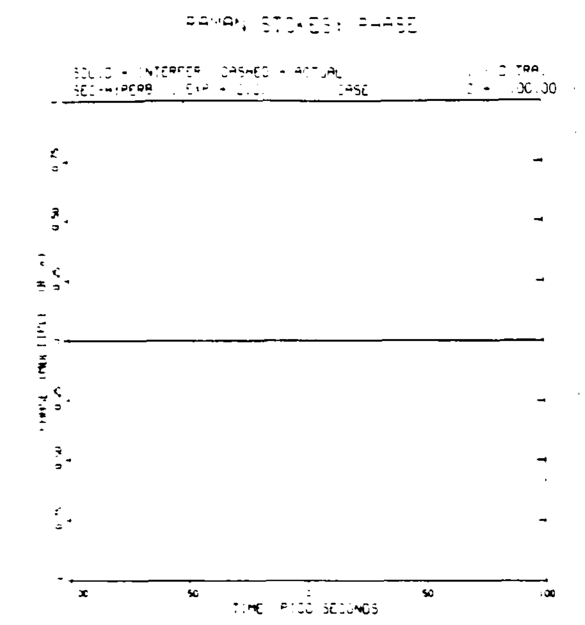


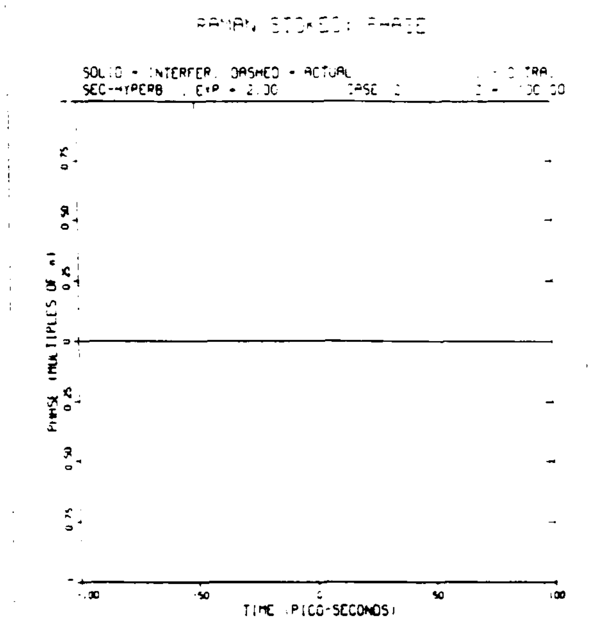


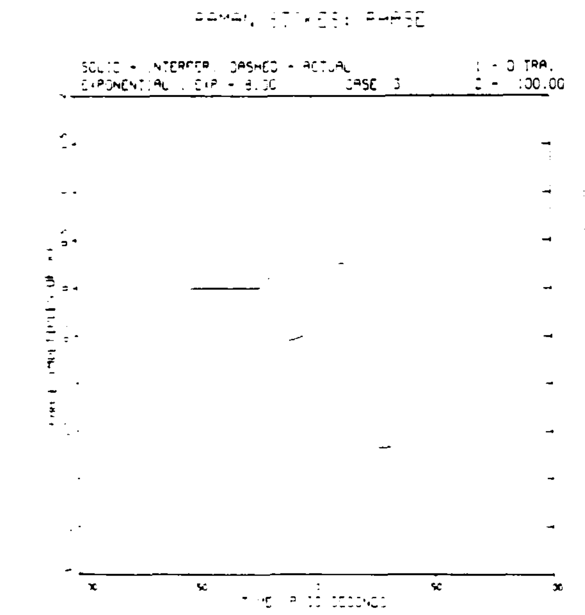


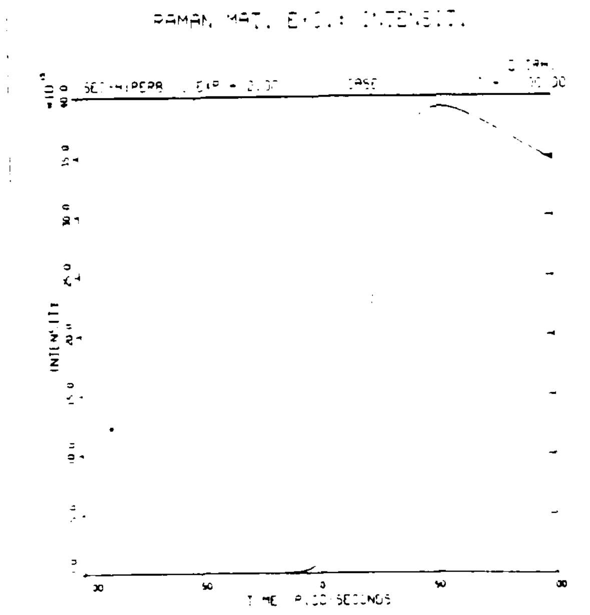


PLT2.DAT (Example A2)

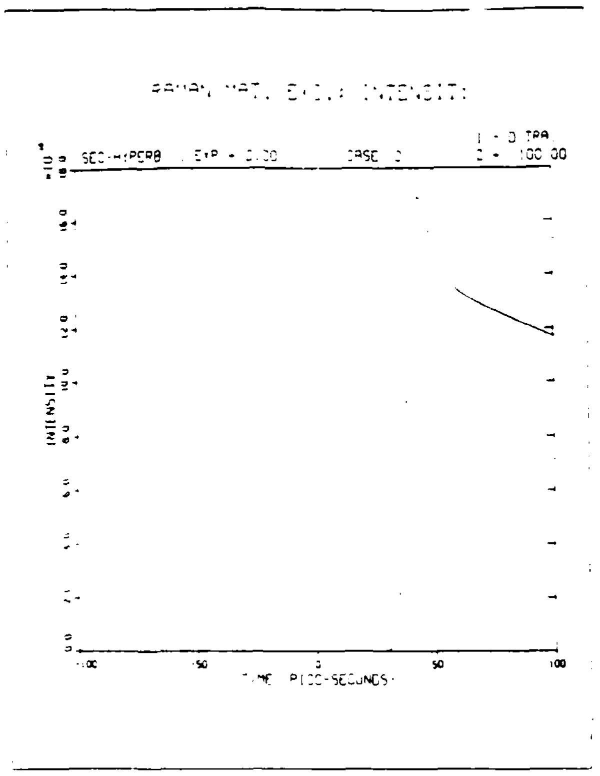


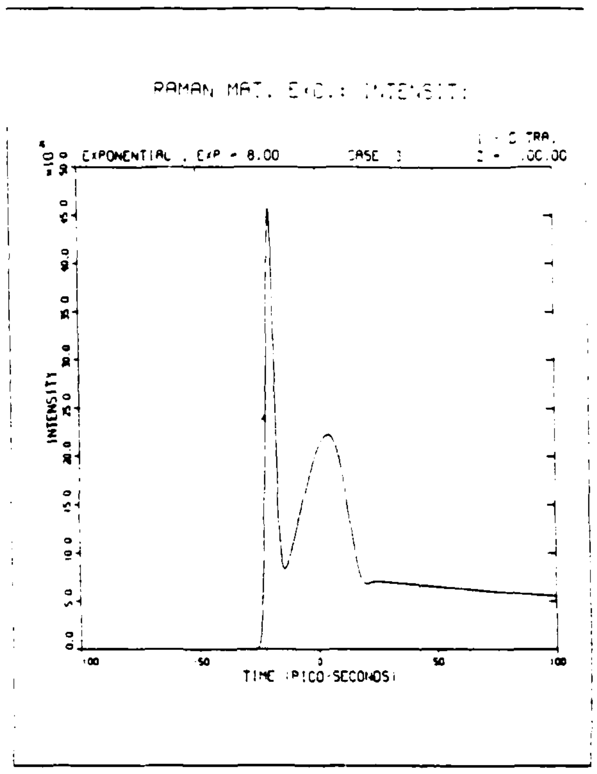


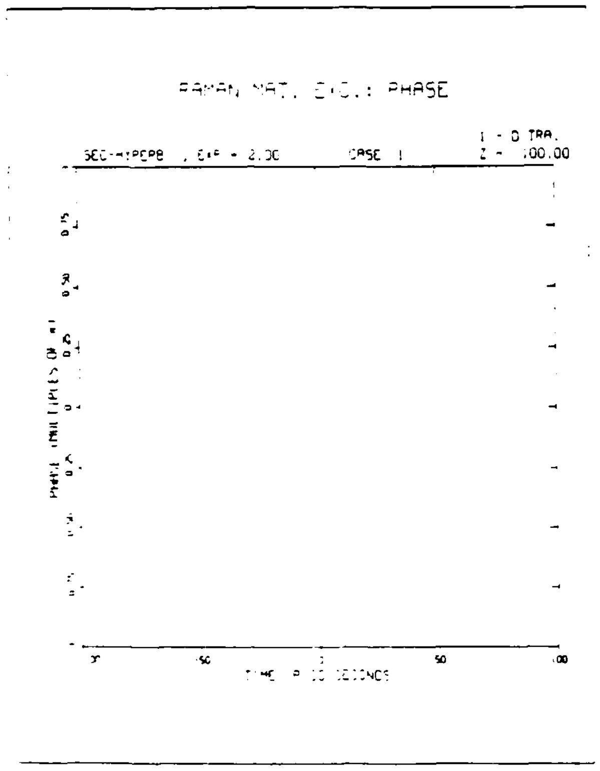


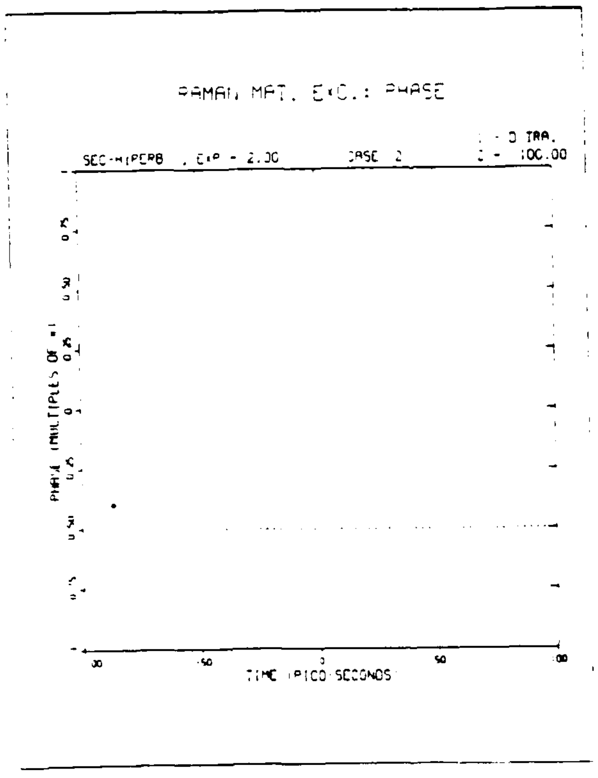


PLT2.DAT (Example A2)

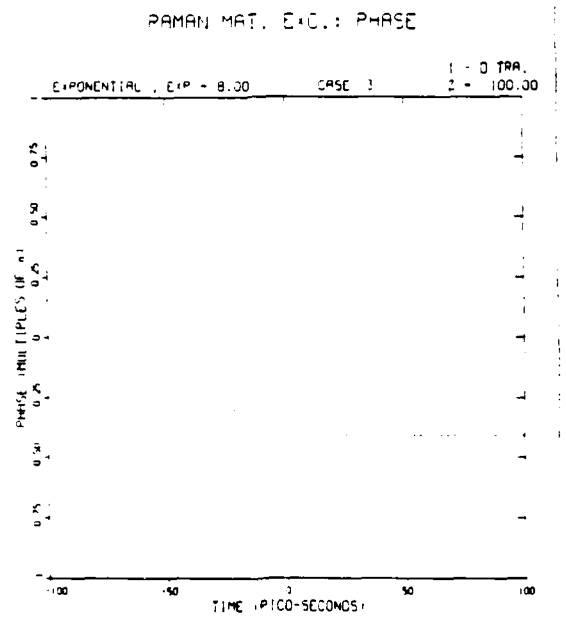








PLT2.DAT (Example A2)



APPENDIX B 1-D Stationary Limit; Examples

Two examples are appended to illustrate code operation in the stationary limit. The illustration features the batch job command files, the input data files, the ouput CPR-files and the resulting output. The first example is a simple run that features a chirped input signal and fast Fourier transforms of the fields. The second example illustrates several cases of multiple aberrated beam interaction in one run of the programs.

EXAMPLE B1

X1J.JOB

```
AUDIT.
FETCH,    DN=NRAM,TEXT='NR1J.DAT'.
ACCESS,   DN=XR1J.
XR1J.
DISPOSE,  DN=ERRM,DF=BB,WAIT,TEXT='X1J.MSG.'.
ACCESS,   DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=DVSD,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=XP1J.
FETCH,    DN=NPRAM1,TEXT='NP1J.DAT'.
XP1J.
AUDIT.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='X1J.MSG.'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='X1J.DSP.'.
EXIT.
```

NR1J.DAT

```
$NAML
  RIST=1.0E-12,
  PHL(1)=3.14,
  PHL(2)=3.14,
  PHST=3.14,
  ICOND=4,
  ZFINAL=40.0,
  ZKEEP=10.0,
  GAIN=0.4,
$
     NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
    1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
    2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
```

NP1J.DAT

```
$FLDATE
  DONYET=1,
  MONTH=03,
  DAY=28,
  YEAR=88,
  IPART=2,
  NEDN=1,
$
$CONDAT
  LPRMT(1)=1,
  LPRMT(2)=1,
  LPRMT(3)=1,
  LPRMT(4)=1,
  NSEC=1,
  CSEC(1,1)=(1.0,2.0),
  CSEC(2,1)=(1.0,2.0),
  CSEC(4,1)=(1.0,2.0),
  CSEC(5,1)=(1.0,2.0),
  CSEC(7,1)=(1.0,2.0),
  CSEC(8,1)=(1.0,2.0),
  CSEC(10,1)=(1.0,2.0),
  CSEC(11,1)=(1.0,2.0),
  CSEC(13,1)=(1.0,2.0),
  CSEC(14,1)=(1.0,2.0),
  CSEC(16,1)=(1.0,2.0),
  CSEC(17,1)=(1.0,2.0),
$
$ZPLOT
  KZ(1)=1,
  KZ(2)=2,
  KZ(3)=3,
  KZ(2)=4,
  KZ(3)=5,
$
```

X1J.CPR

```
16:30:55.8536   0.0000  - CSP   ....................................................................
16:30:55.8539   0.0000    CSP   .                                                                  .
16:30:55.8541   0.0000    CSP   .                    WELCOME TO THE NRL CRAY XMP                   .
16:30:55.8544   0.0000    CSP   .                                                                  .
16:30:55.8546   0.0001    CSP   ....................................................................
16:30:55.8549   0.0001    CSP   .    We are attempting to cleanup the library of CRAY archive tapes .
16:30:55.8551   0.0001    CSP   .    These  cleanup' runs will be made on Tuesday and Wednesday     .
16:30:55.8554   0.0001    CSP   .    mornings between 2:00 and 7:00 AM   During these times. there  .
16:30:55.8557   0.0001    CSP   .    will be no recall of off line data sets   If you plan on running.
16:30:55.8559   0.0001    CSP   .    jobs during these hours. please insure that required files are .
16:30:55.8562   0.0001    CSP   .    on line                                                        .
16:30:55.8564   0.0001    CSP   ....................................................................
16:30:56.0923   0.0002    CSP        CRAY X MP SERIAL 415 65    NAVAL RESEARCH LABORATORY    03 28 88
16:30:56.0926   0.0002    CSP
16:30:56.0950   0.0002    CSP        CRAY OPERATING SYSTEM          COS 1 15  ASSEMBLY DATE 01 04 88
16:30:56.0954   0.0002    CSP
16:30:56.0959   0.0002    CSP
16.30.56 1127   0.0002    CSP    JOB.JN=X1J.MFL=511000.US=DEFER.
16:30:56 1482   0.0013    CSP    ACCOUNT.AC-.US-.UPW-.APW-
16:30:58 6049   0 1023    USER  AC213   '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
16:30:59 0474   0 1051    USER  AUDIT
16 31:03 6942   0 1872    USER  AU003         63 DATASETS.    95112 BLOCKS.    48671617 WORDS
16:31:03 6946   0 1873    USER  AU003 - '      6 DATASETS.     1535 BLOCKS.      784380 WORDS ONLINE
16:31:03 6950   0.1874    USER  AU003 -       57 DATASETS.    93577 BLOCKS.    47887237 WORDS OFFLINE
16:31:03 7015   0 1875    CSP   FETCH.   DN=NRAM.TEXT= NR1J DAT'.
16:31:09 1017   0 1876    SCP    VAX TO CRAY: %SYSTEM S NORMAL. normal successful completion
16 31 09 1020   0 1876    SCP    VAX TO CRAY: FILE-$1$DUA107:[HILFER FR2]NR1J DAT:18
16:31:09 1022   0 1876    SCP    VAX TO CRAY: 488 BYTES TRANSFERRED
16:31:12 8095   0 1876    SCP   SS004 - DATASET RECEIVED FROM FRONT END
16 31:12 9678   0.1878    CSP   ACCESS.  DN=XR1J.
16 31:13 0703   0 1878    PDM   PD000 - PDN = XR1J              ID =            ED =     4  OWN = HILFER
16 31:13 0706   0 1878    PDM   PD000 - ACCESS  COMPLETE
16 31 13 1357   0 1880    CSP   XR1J
16 31 19 9470   5 8062    PDM   PD000 - PDN = F1J032888         ID =            ED =     1  OWN = HILFER
16 31 19 9472   5 8062    PDM   PD000 - SAVE     COMPLETE
16 31 19 9489   5 8062    USER  UT003 - EXIT CALLED BY  RAM2D1C
16:31:19 9540   5 8062    CSP   DISPOSE. DN=ERRM.DF=BB.WAIT.TEXT='X1J MSG.'.
16:31:31 1975   5 8064    SCP    CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
16 31 31 1978   5 8064    SCP    CRAY TO VAX: FILE-$1$DUA107:[HILFER FR2]X1J MSG;3
16 31 31 1982   5 8064    SCP    CRAY TO VAX: 51004 BYTES TRANSFERRED
16:31:39 1423   5 8067    CSP   ACCESS.  DN=DISLIB.ID=DISSPLA.OWN=LIBRARY.
16:31:39 6518   5 8068    PDM   PD000 - PDN = DISLIB            ID = DISSPLA    ED =     1  OWN = LIBRARY
16:31:39 6520   5 8068    PDM   PD000 - ACCESS  COMPLETE
16.31.39 6541   5 8071    CSP   ACCESS.  DN=INTLIB.ID=DISSPLA.OWN=LIBRARY.
16:31.40 2745   5 8072    PDM   PD000 - PDN = INTLIB            ID = DISSPLA    ED =     1  OWN = LIBRARY
16 31 40 2748   5 8072    PDM   PD000   ACCESS  COMPLETE
16:31:40 2768   5 8075    CSP   ACCESS.  DN=DVSD.ID=DISSPLA.OWN=LIBRARY
16 31:40 5113   5 8075    PDM   PD000 - PDN = DVSD              ID = DISSPLA    ED =     1  OWN = LIBRARY
16 31:40 5116   5 8075    PDM   PD000 - ACCESS  COMPLETE
16 31 40 5132   5 8077    CSP   ACCESS.  DN=XP1J
16 31 40 7760   5 8077    PDM   PD000 - PDN = XP1J              ID =            ED =     5  OWN = HILFER
16 31 40 7763   5 8077    PDM   PD000   ACCESS  COMPLETE
16:31:40 7778   5 8078    CSP   FETCH.   DN=NPRAM1.TEXT= NP1J DAT'
16:31:45 0963   5 8079    SCP    VAX TO CRAY: %SYSTEM-S-NORMAL. normal successful completion
16.31:45 0966   5 8079    SCP    VAX TO CRAY: FILE-$1$DUA107:[HILFER FR2]NP1J.DAT:19
16.31:45 0968   5.8079    SCP    VAX TO CRAY: 912 BYTES TRANSFERRED
16 31 48 2089   5 8079    SCP   SS004 - DATASET RECEIVED FROM FRONT END
16 31 48 4865   5 8081    CSP   XP1J
16:31:48 9837   5 8117    PDM   PD000 - PDN = F1J032888         ID =            ED =     1  OWN = HILFER
16:31 48 9840   5 8117    PDM   PD006 - LOCAL DATASET NAME ALREADY IS IN USE
```

X1J.CPR

```
16 32 00 8542       15 0458    USER    UT003    EXIT CALLED BY   PRAMICD
16.32:01 0706       15 0462    USER    AUDIT
16 32 05 8121       15 1294    USER    AU003          64 DATASETS.     95257 BLOCKS.     48745387 WORDS
16 32 05 8125       15 1295    USER    AU003           7 DATASETS.      1680 BLOCKS.       858100 WORDS ONLINE
16 32 05 8129       15 1296    USER    AU003          57 DATASETS      93577 BLOCKS      47887287 WORDS OFFLINE
16 32 05 8198       15 1296    CSP     DISPOSE. DN-META.DF=BB.WAIT.TEXT= PLT2 DAT
16 32 19 7106       15 1298    SCP      CRAY TO VAX: %RMS-S-NORMAL. normal successful completion
16 32.19 7109       15 1298    SCP      CRAY TO VAX. FILE-$1$DUA107.[HILFER.FR2]PLT2.DAT.3
16 32 19 7111       15 1298    SCP      CRAY TO VAX: 391680 BYTES TRANSFERRED
16 32 21 9142       15 1299    CSP     DISPOSE. DN EPRM.DF BB.WAIT.TEXT 'X1J MSG
16 32:26 0557       15 1300    SCP      CRAY TO VAX: %RMS-S-NORMAL  normal successful completion
16 32:26 0560       15 1300    SCP      CRAY TO VAX: FILE-$1$DUA107:[HILFER.FR2]X1J MSG.4
16:32 26 0562       15 1300    SCP      CRAY TO VAX: 24879 BYTES TRANSFERRED
16.32 27 6037       15 1301    CSP     DISPOSE. DN-DISOUT.DF-BB.WAIT.TEXT  X1J DSP
16 32 33 0014       15 1303    SCP      CRAY TO VAX: %RMS S NORMAL. normal successful completion
16 32 33 0017       15 1303    SCP      CRAY TO VAX: FILE $1$DUA107:[HILFER FR2]X1J DSP.2
16·32·33 0019       15 1303    SCP      CRAY TO VAX: 1004 BYTES TRANSFERRED
16 32 34 7496       15 1303    CSP     EXIT
16:32:34 7510       15 1304    CSP     END OF JOB
16:32:34 7513       15 1304    CSP
16 32.34 7516       15 1304    CSP
16 32 35 2206       15 1305    USER          JOB NAME                                  X1J
16:32·35 2209       15 1305    USER          USER NUMBER -                             HILFER
16·32 35 2413       15 1305    USER          JOB SEQUENCE NUMBER -                           34182
16 32:35 2418       15 1305    USER
16 32:35 2422       15 1305    USER          TIME EXECUTING IN CPU -                   0000:00:15 1305
16 32 35 2426       15 1306    USER          TIME WAITING TO EXECUTE -                 0000:01:10 4314
16 32:35 2429       15 1306    USER          TIME WAITING FOR I O                      0000:00:13.2222
16·32:35 2432       15 1306    USER          TIME WAITING IN INPUT QUEUE -             0000:00:00 0021
16.32:35 2435       15 1306    USER          MEMORY * CPU TIME (MWDS*SEC) -                  2.74438
16·32:35 2438       15 1306    USER          MEMORY * I O WAIT TIME (MWDS*SEC) -             1.72518
16.32.35 2442       15 1306    USER          MINIMUM JOB SIZE (WORDS) -                 44544
16 32 35 2445       15 1306    USER          MAXIMUM JOB SIZE (WORDS)                  228864
16 32:35 2448       15 1307    USER          MINIMUM FL (WORDS)                         40960
16·32·35 2451       15 1307 -  USER          MAXIMUM FL (WORDS)                        224256
16 32:35 2454       15 1307    USER          MINIMUM JTA (WORDS)                         3584
16:32:35 2457       15.1307    USER          MAXIMUM JTA (WORDS)                         5632
16 32 35 2460       15 1307    USER          DISK SECTORS MOVED                          3305
16 32.35 2463       15 1307    USER          FSS SECTORS MOVED                              0
16 32 35 2466       15 1307    USER          USER I O REQUESTS                            915
16·32:35 2469       15 1307    USER          USER I O SUSPENSIONS                        1308
16 32.35 2473       15 1307    USER          OPEN CALLS -                                  35
16.32:35 2476       15.1307    USER          CLOSE CALLS                                   34
16·32·35 2479       15 1307    USER          MEMORY RESIDENT DATASETS -                     0
16 32:35 2482       15 1307    USER          TEMPORARY DATASET SECTORS USED               145
16 32 35 2485       15 1308    USER          PERMANENT DATASET SECTORS ACCESSED          2810
16 32·35 2488       15 1308    USER          PERMANENT DATASET SECTORS SAVED              145
16 32 35 2491       15 1308    USER          SECTORS RECEIVED FROM FRONT END -              2
16·32.35 2494       15 1308    USER          SECTORS QUEUED TO FRONT END                  144
16 32:35 5314       15 1380    USER
16:32 35 5317       15 1380    USER    ..................................................................................
16 32 35 5320       15 1380    USER               ***   COST TABLE FOR THIS JOB ***
16·32·35 5323       15 1381    USER                       JOBNAME                                      X1J
16·32·35 5327       15 1382    USER                       USER IDENT ---------                         HILFER
16 32:35 5330       15 1383    USER                       BEGAN EXECUTION       MON MAR 28. 1988       16:30:55  HOURS
16 32 35 5333       15 1384    USER                       AT A PRIORITY OF  --                               3
16 32 35 5337       15 1385    USER                       AND JOB CLASS OF                             DSMALL
16 32 35 5340       15 1386    USER            15 136789  SECONDS OF CPU TIME     @ $ 630 00  HR            $        2 65
16 32 35 5344       15 1387    USER             2 744818  MEMORY*CPU (MWRD SEC)   @ $  84 00  HR            $        0 06
16 32 35 5347       15 1388    USER             1 728561  MEMORY*I O (MWRD-SEC)   @ $  84 00  HR       --   $        0 04

16 32 35 5351       15 1389    USER             0 003307  I O MEGASECTORS MOVED   @ $  84 00  EA            $        0 28
16 32 35 5355       15 1390    USER             0 000000  TAPE MOUNT(S)           @ $   5 00  EA            $        0 00
16 32 35 5412       15 1391    USER
16 32 35 5414       15 1391    USER                       ****    TOTAL COST FOR THIS JOB     ****          $        3 03
16 32 35 5416       15 1391    USER    ..................................................................................
```

PLT2.DAT (Example B1)

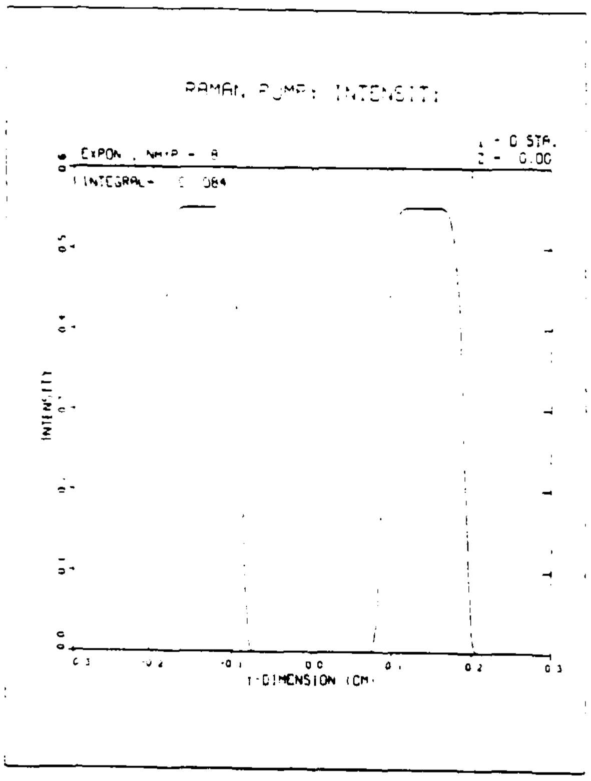
```
                LIST OF INPUT PARAMETERS

  ICOND    -     4
  NHYP     -     8
  NMAX     -     4000
  NPUMP    -     2
  NT       -     1
  NT       -     1024
  GAIN     -      0.4000
  RIST     -      1.00*10^[illegible]
  RKP      -      1.18*10^[illegible]
  RKS      -      9.19*10^[illegible]
  TTWO     -      633.00
  YOST     -      0.0000
  THST     -      0.1000
  ZFINAL   -      40.000
  ZINT     -      20.000
  ZKEEP    -      10.000
  ZSTEP    -      0.0500
```

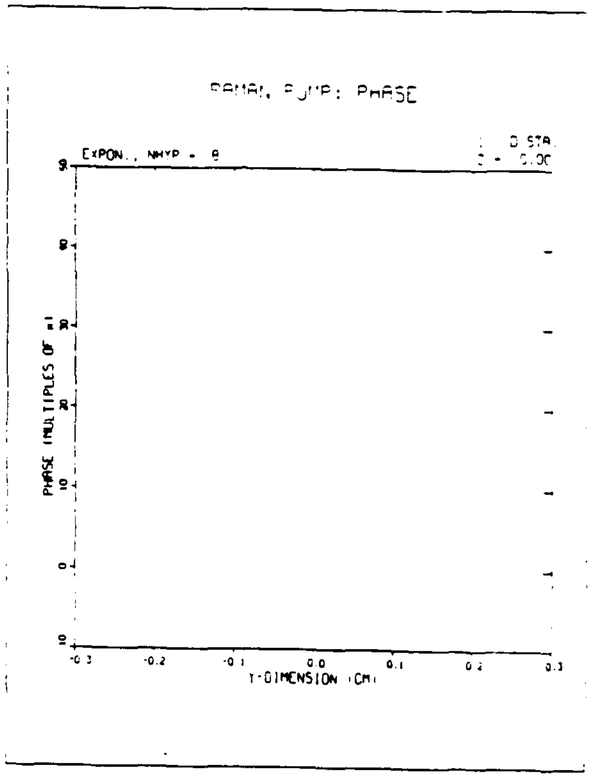
```
  LIST OF INPUT PARAMETERS  CONTD

  RABAMP(1-8) -     0.0000   0.0000   0.0000   0.0000   0.0000
                    0.0000   0.0000   0.0000
  ROSLIM(1-8) -     1.0000   1.0000   1.0000   1.0000   1.0000
                    1.0000   1.0000   1.0000
  RINT(1-10)  -     0.5500   0.5500   0.5500   0.5500   0.5500
                    0.5500   0.5500   0.5500   0.5500   0.5500
  YOFF(1-10)  -     0.1400  -0.1400   0.0000   0.0000   0.0000
                    0.0000   0.0000   0.0000   0.0000   0.0000
  YM(1,2)     -    -0.3000   0.3000
  YWIDTH      -     0.1000   0.1000   0.1000   0.1000   0.1000
                    0.1000   0.1000   0.1000   0.1000   0.1000
```

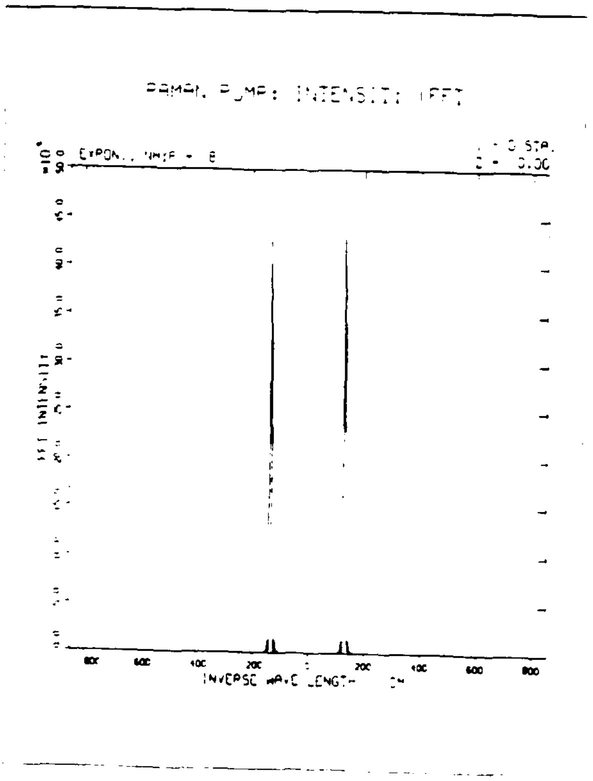
```
  LIST OF INPUT PARAMETERS (CONTD)

  CSEC(1-19,1-8) -
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    1.000  2.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
    0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
```

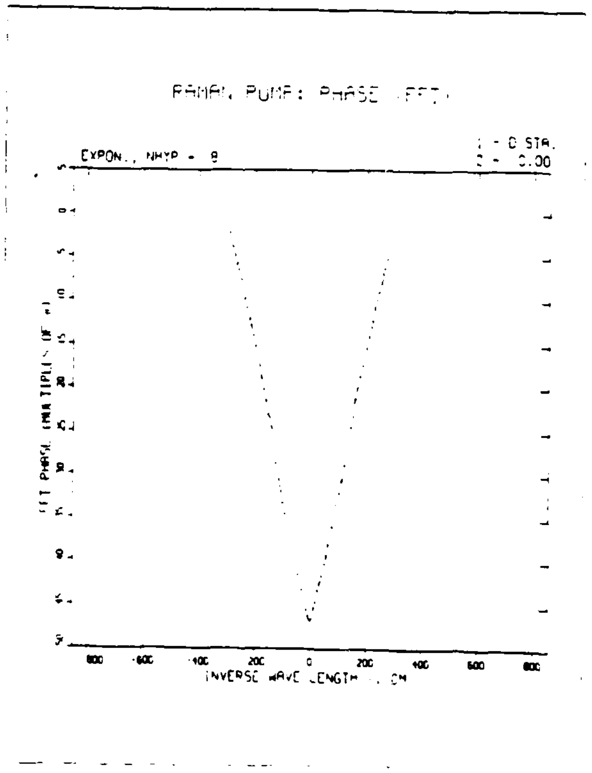
```
                LIST OF OUTPUT PARAMETERS

     PUMP        TOTAL INTENSITY          K-WIDTH

      2          3.82*10^[illegible]        6.18
      1          3.82*10^[illegible]        6.18
```

PLT2.DAT (Example B1)

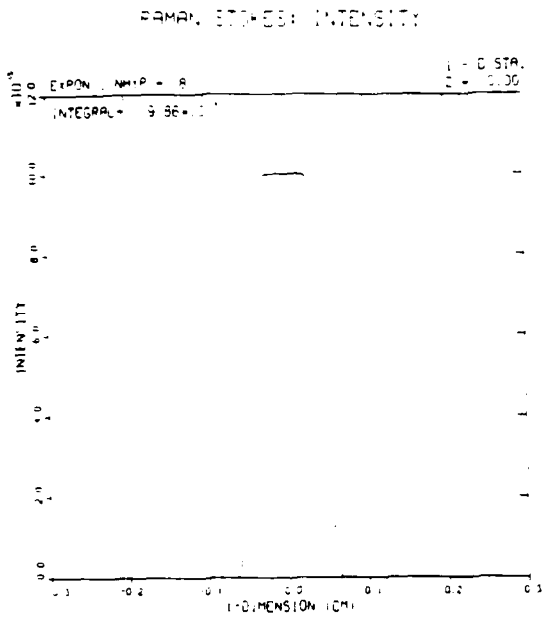


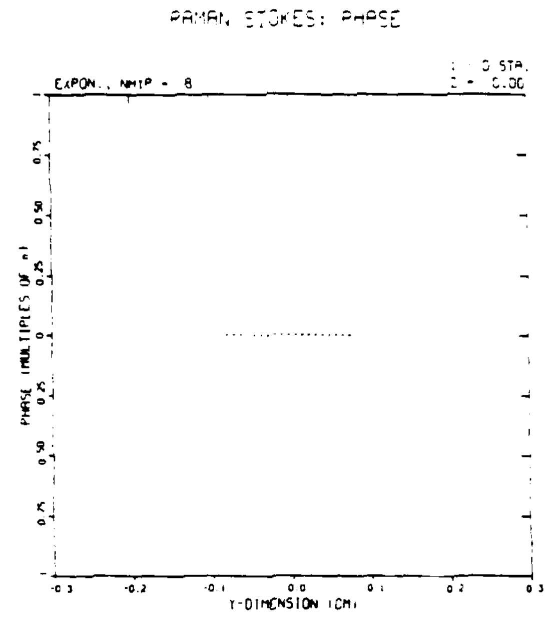


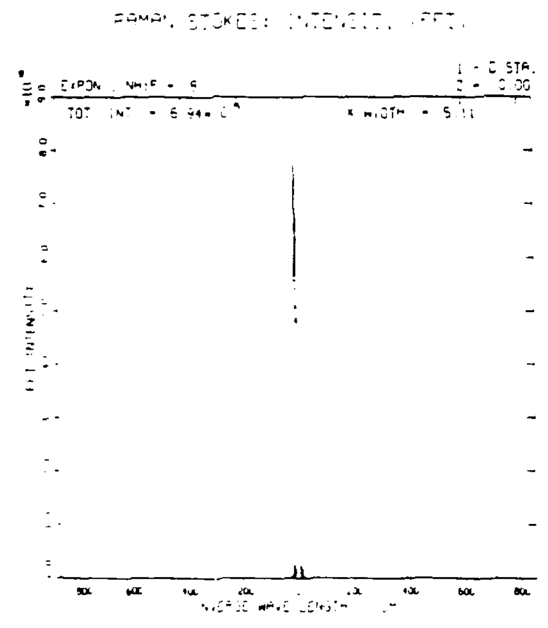


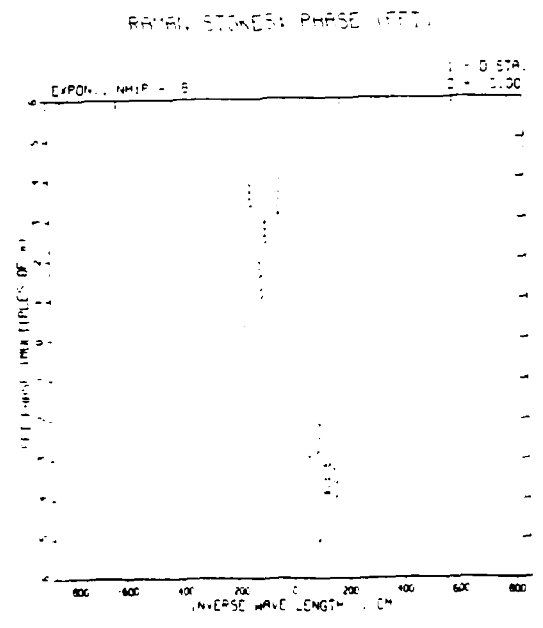


PLT2.DAT (Example B1)

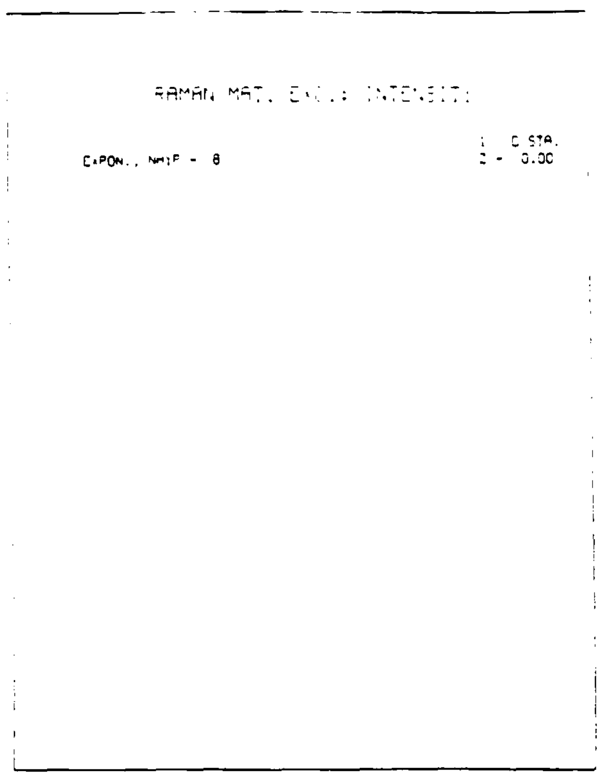


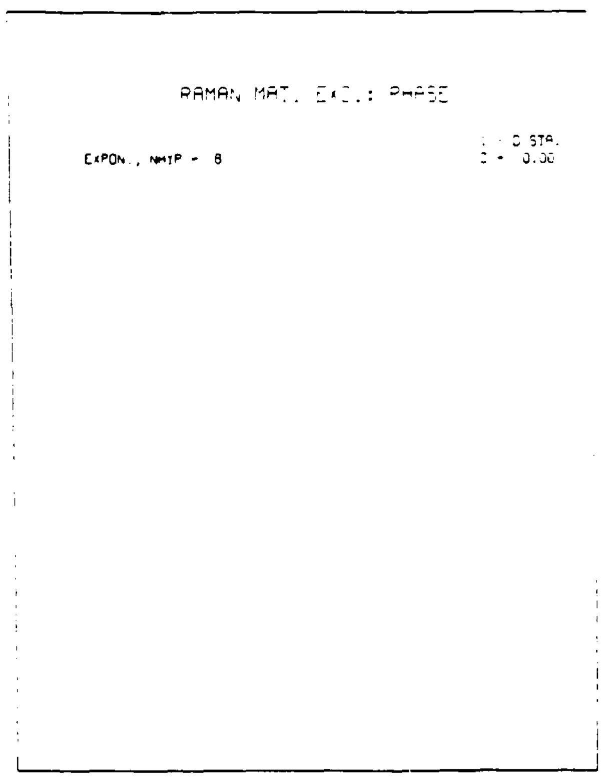


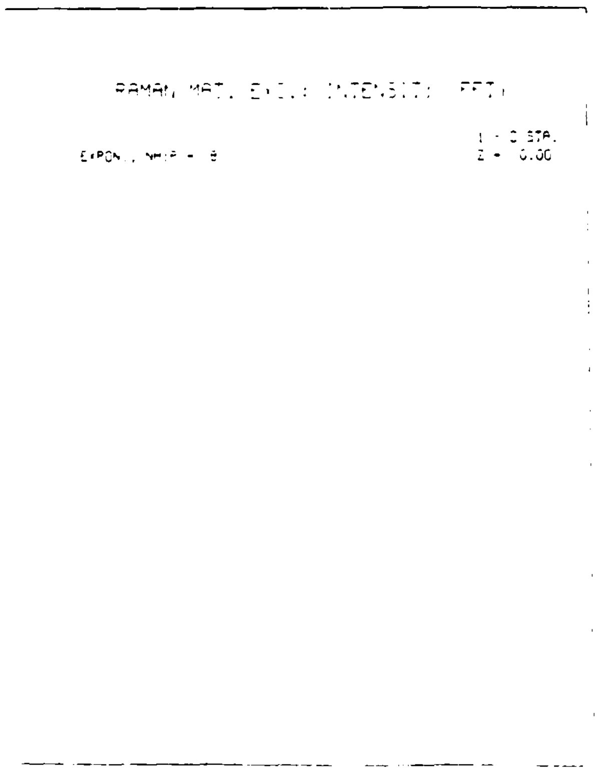


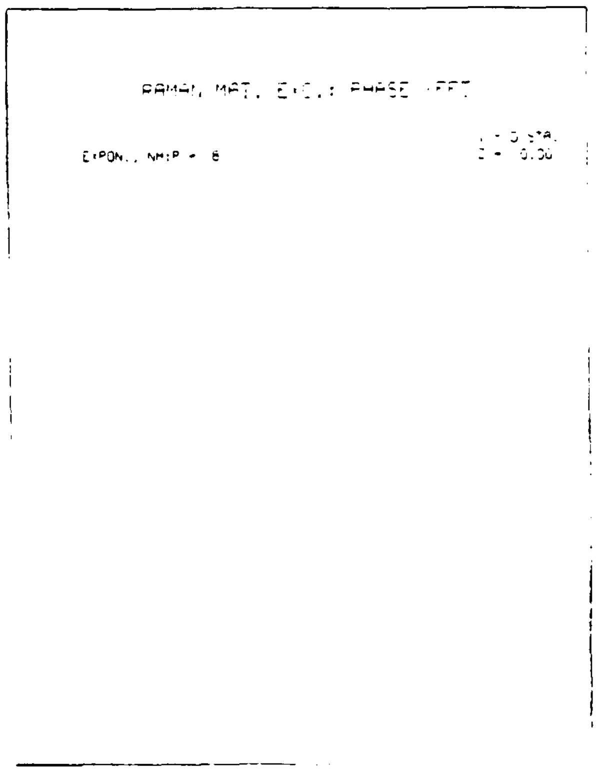


PLT2.DAT (Example B1)

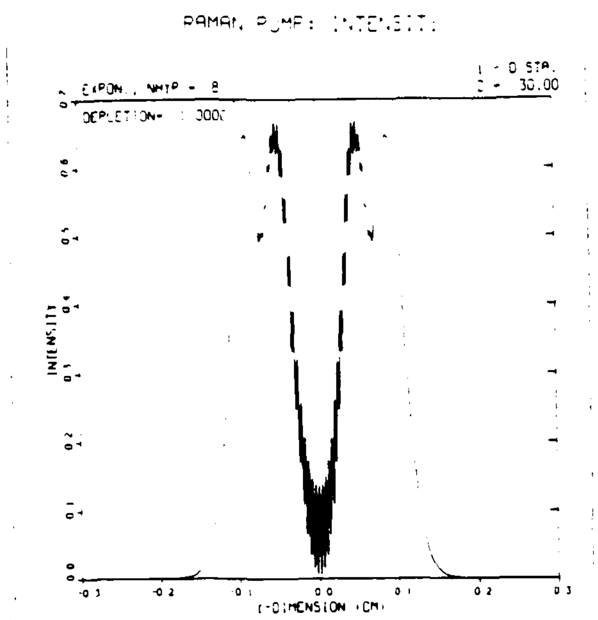


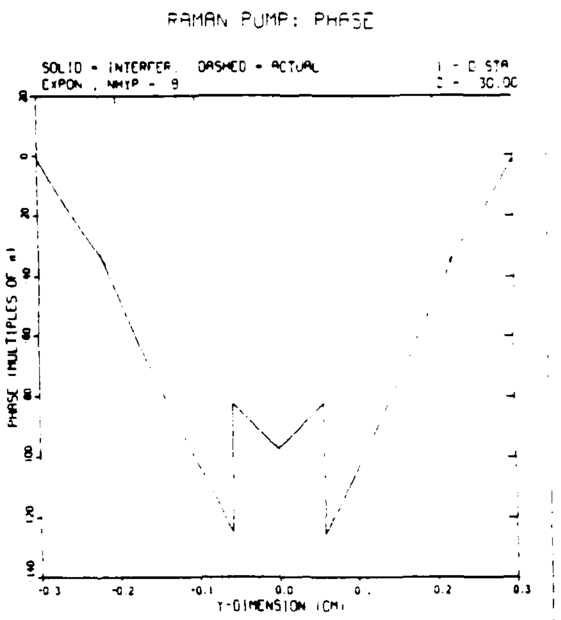


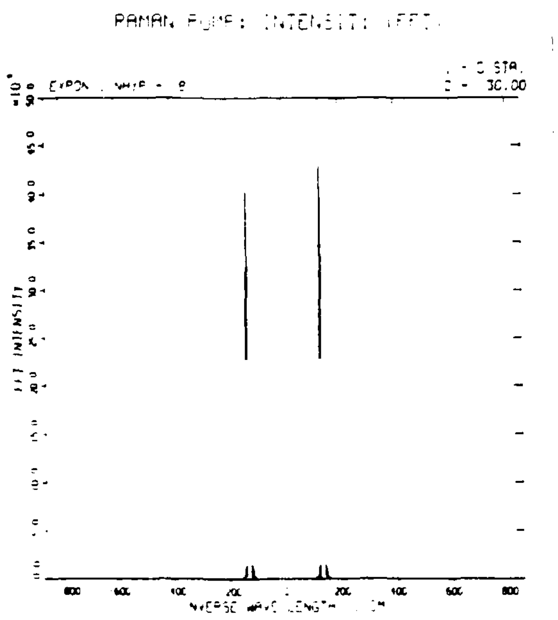


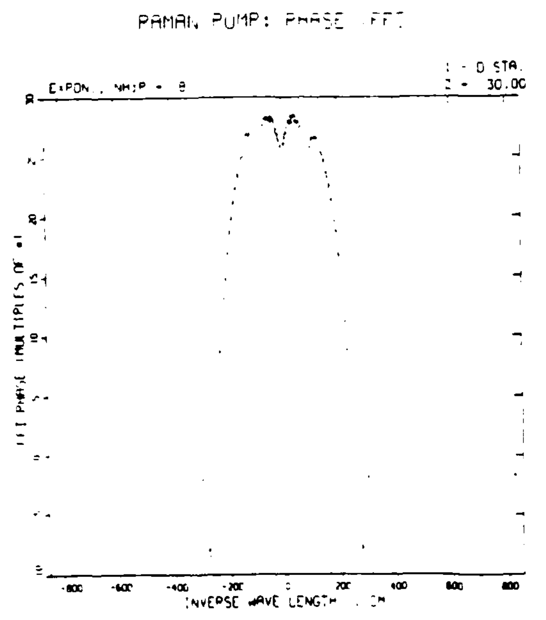


PLT2.DAT (Example B1)

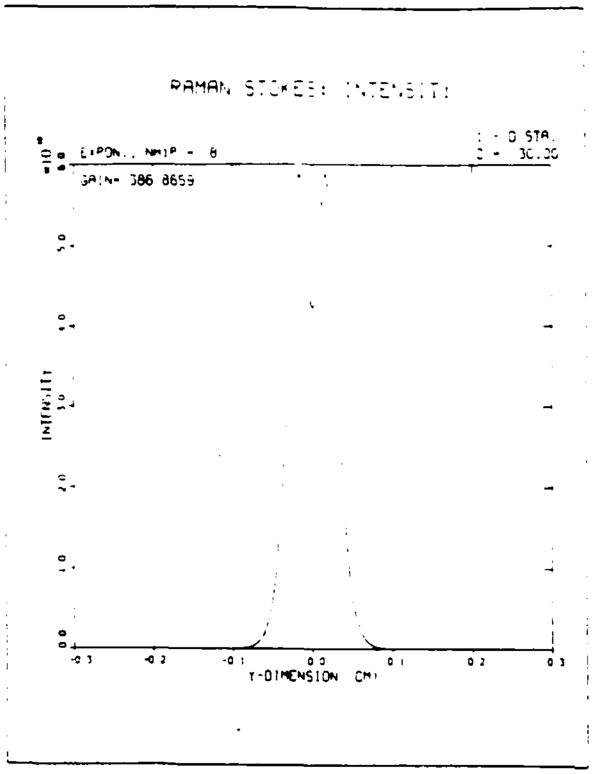


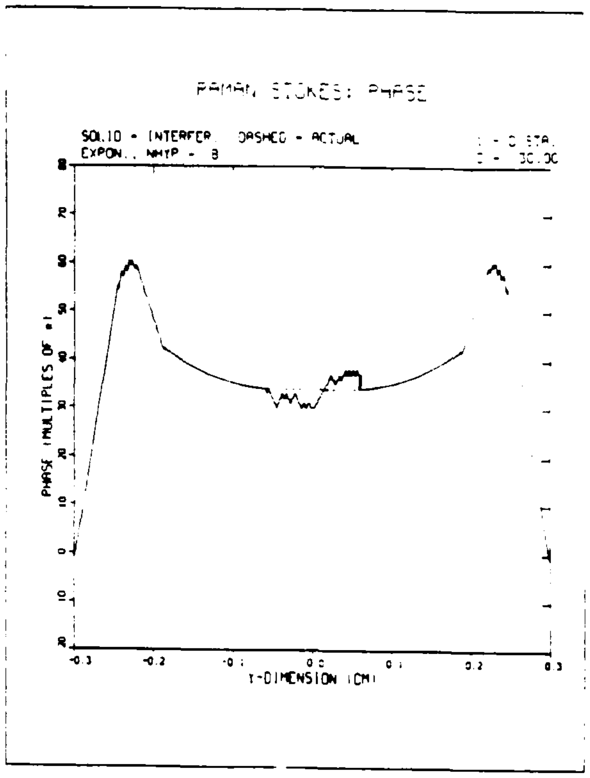


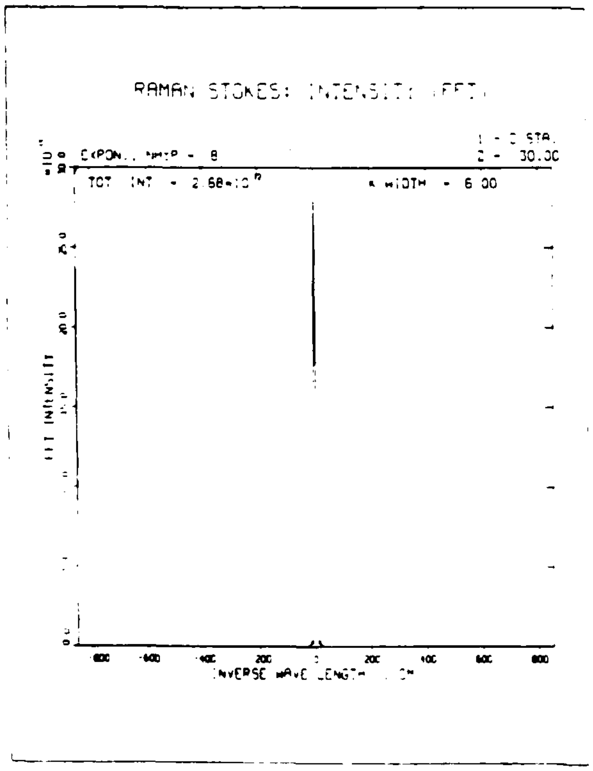


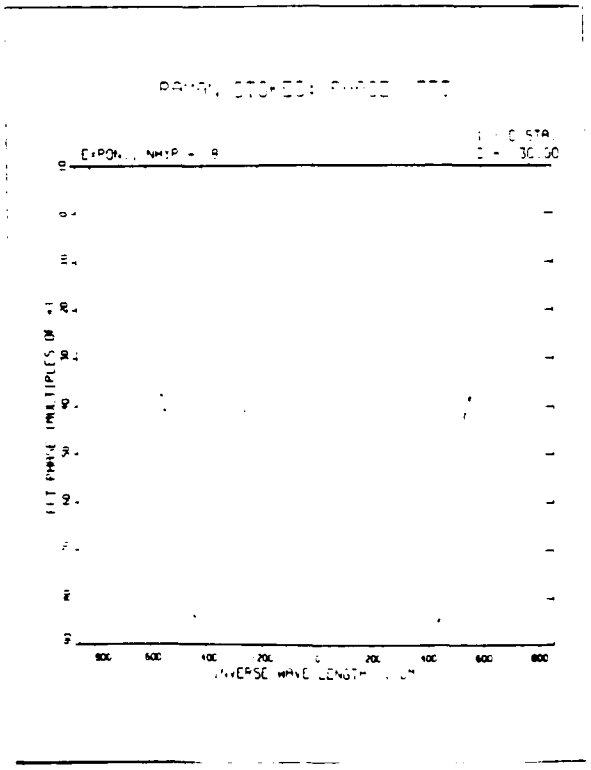


PLT2.DAT (Example B1)

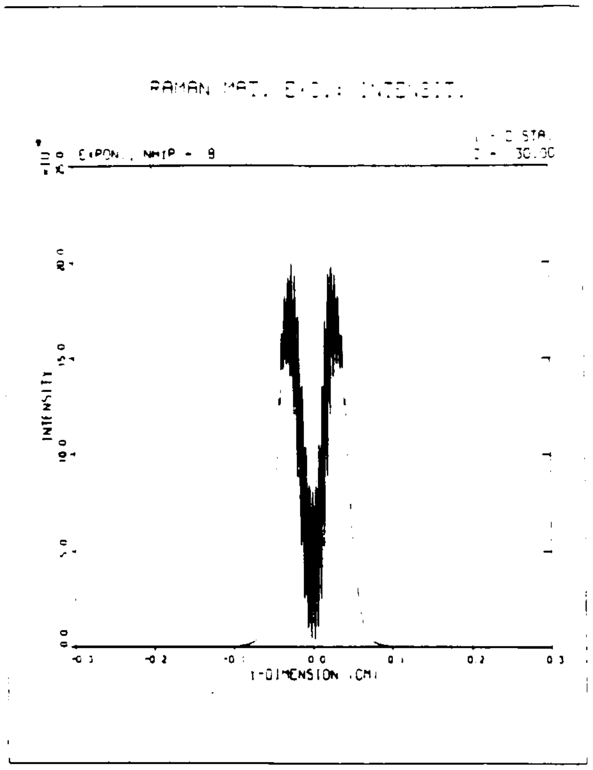


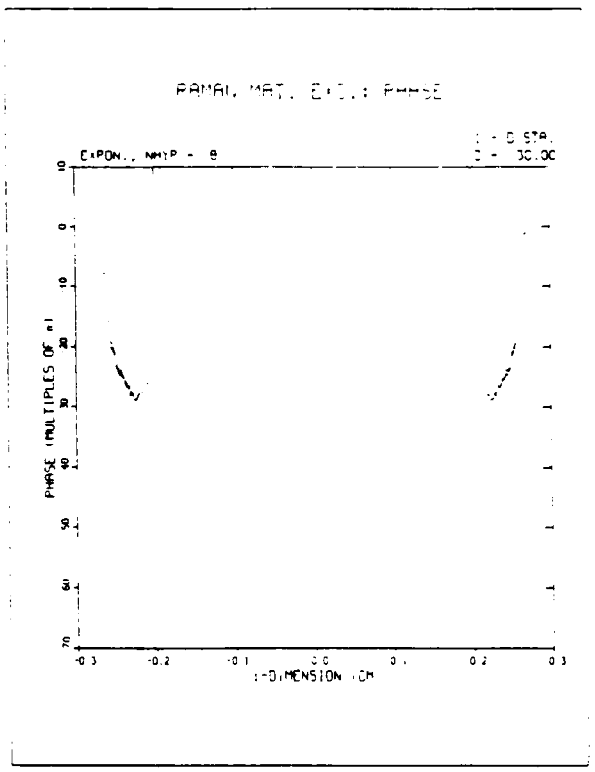


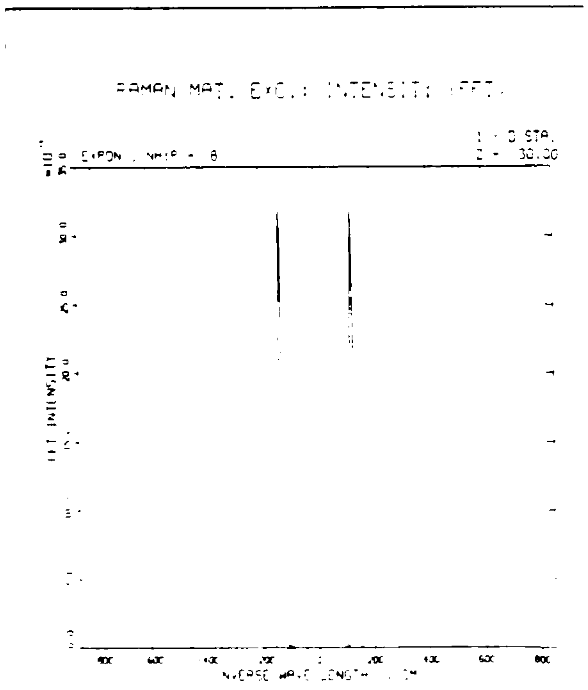


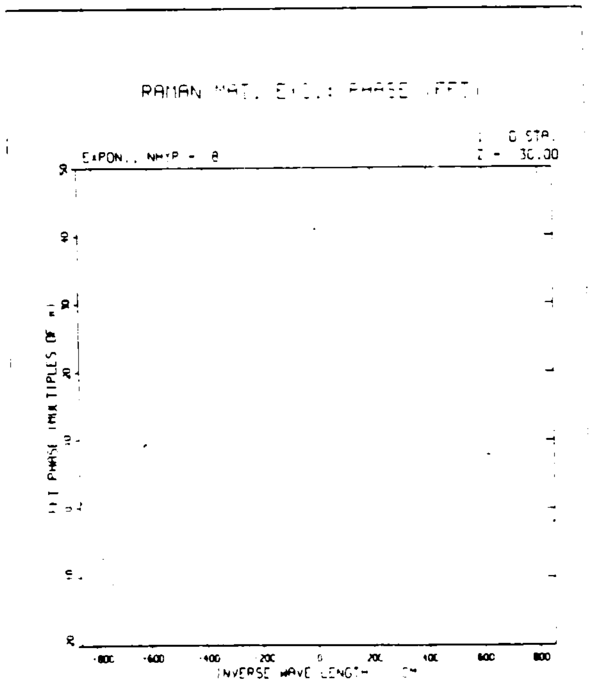


PLT2.DAT (Example B1)

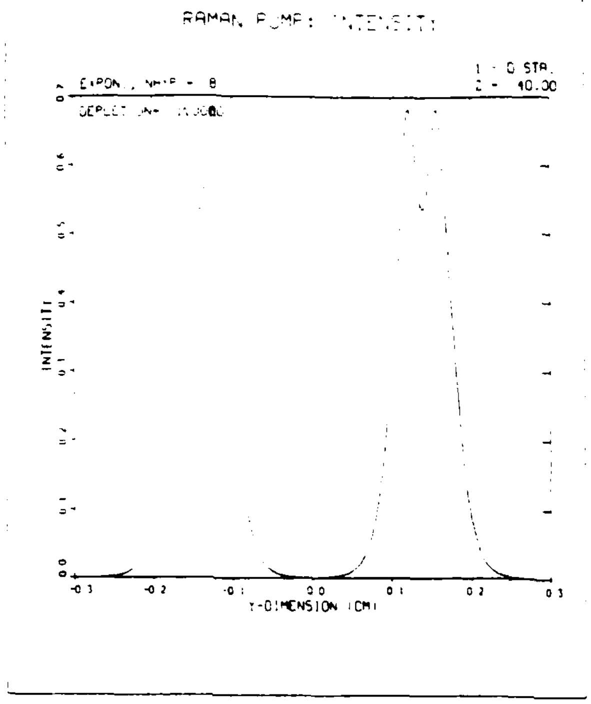


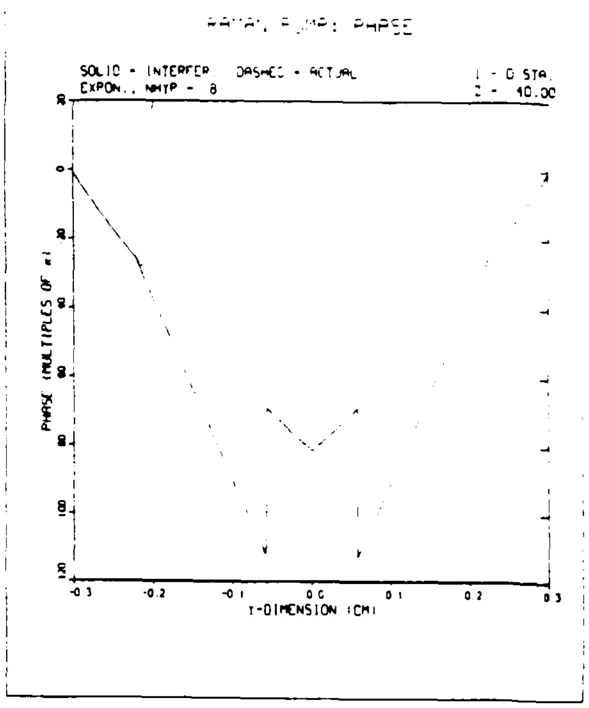


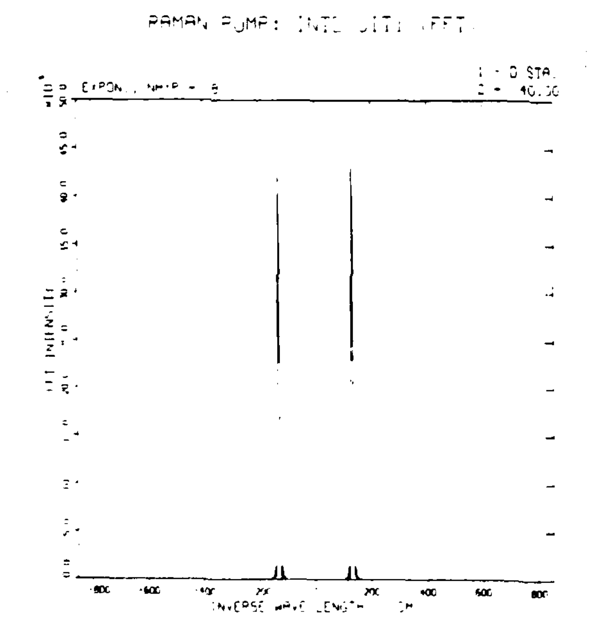


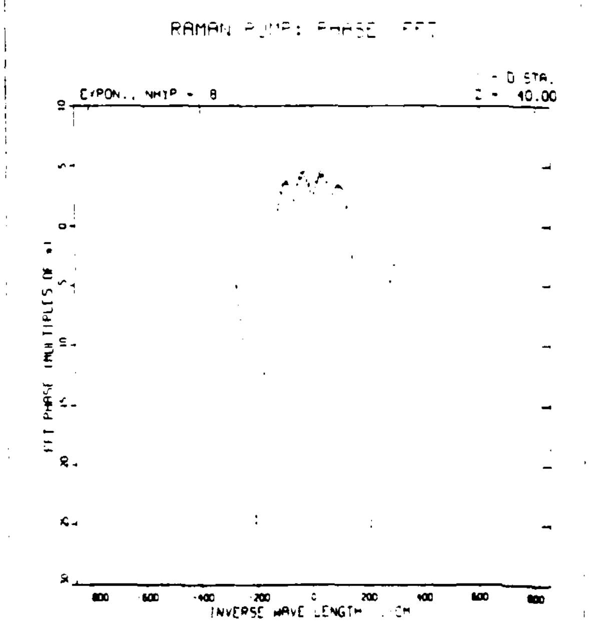


PLT2.DAT (Example B1)

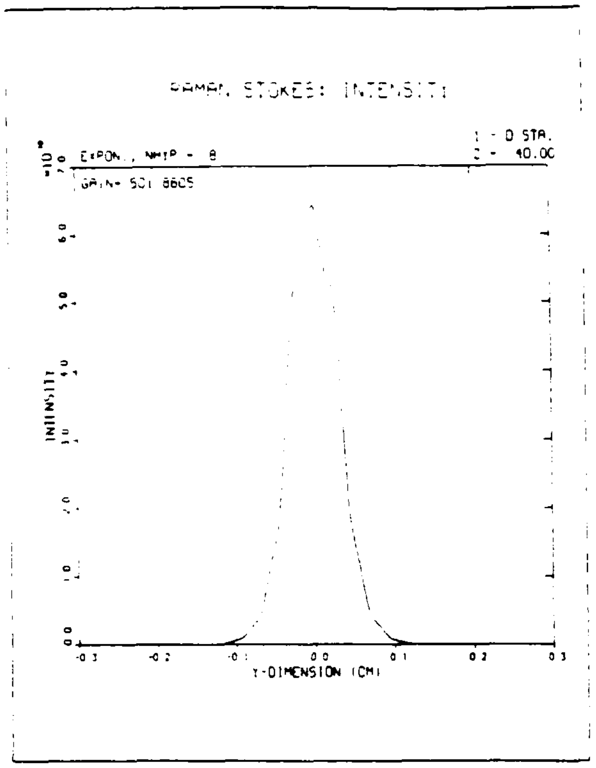


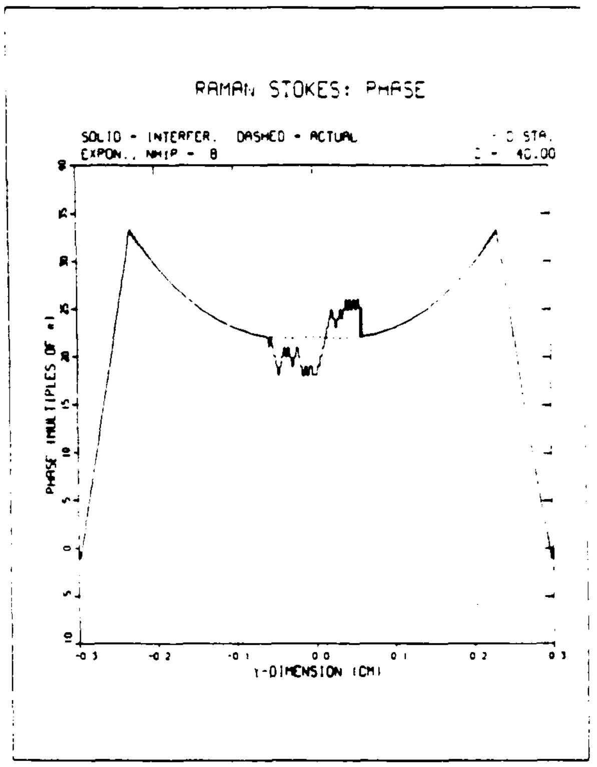


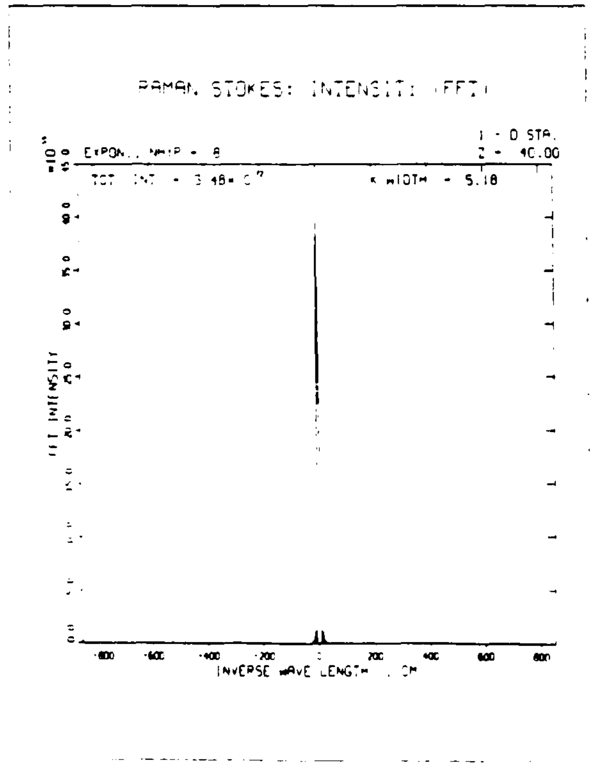


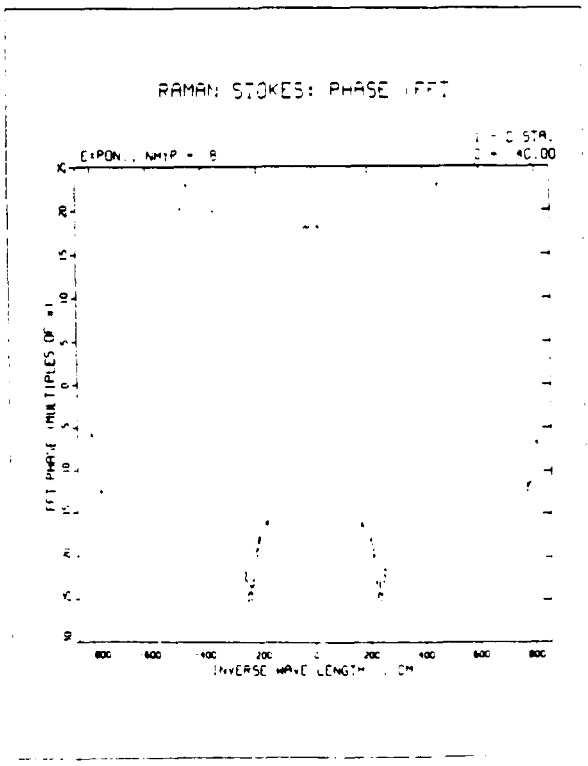


PLT2.DAT (Example B1)

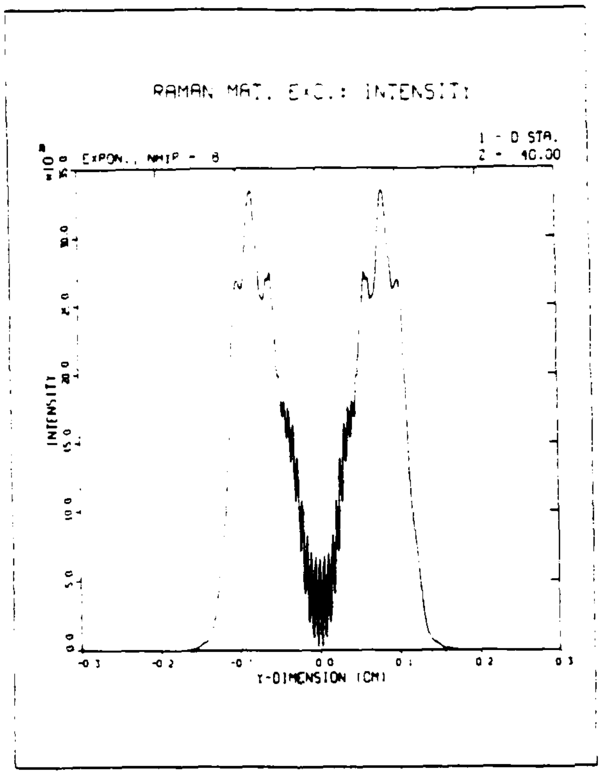


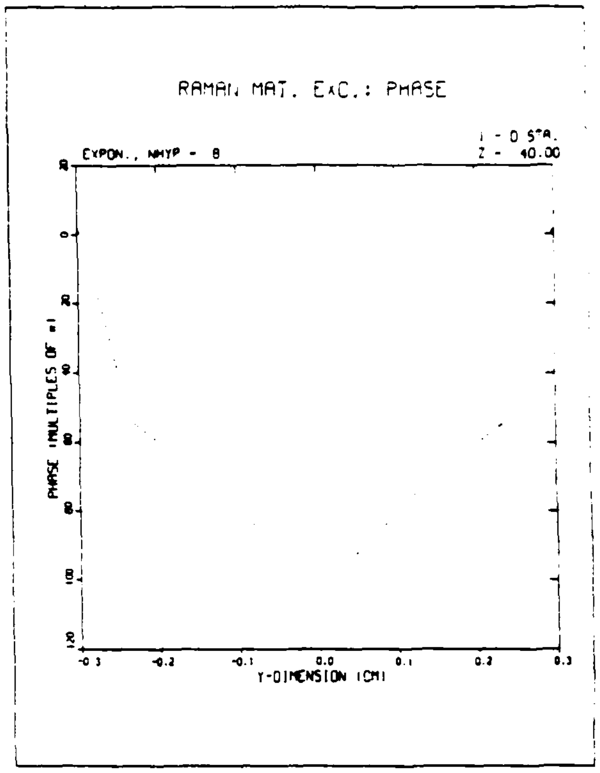


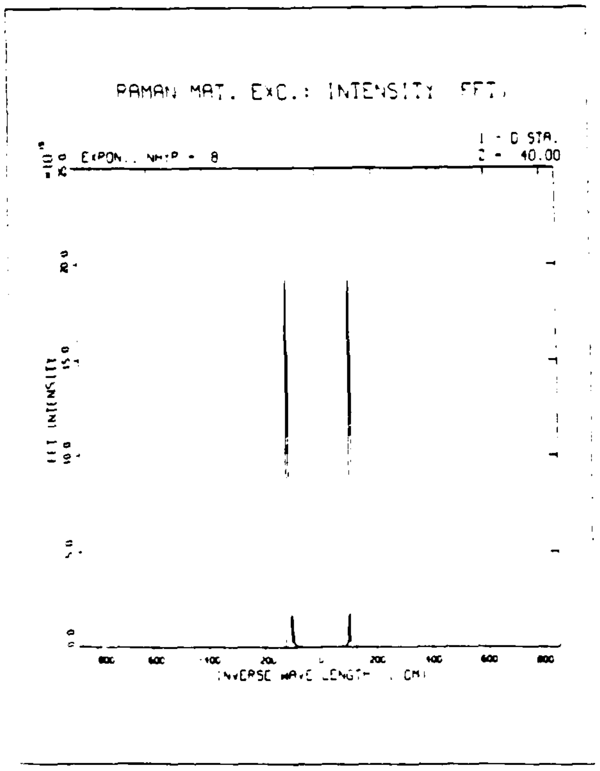


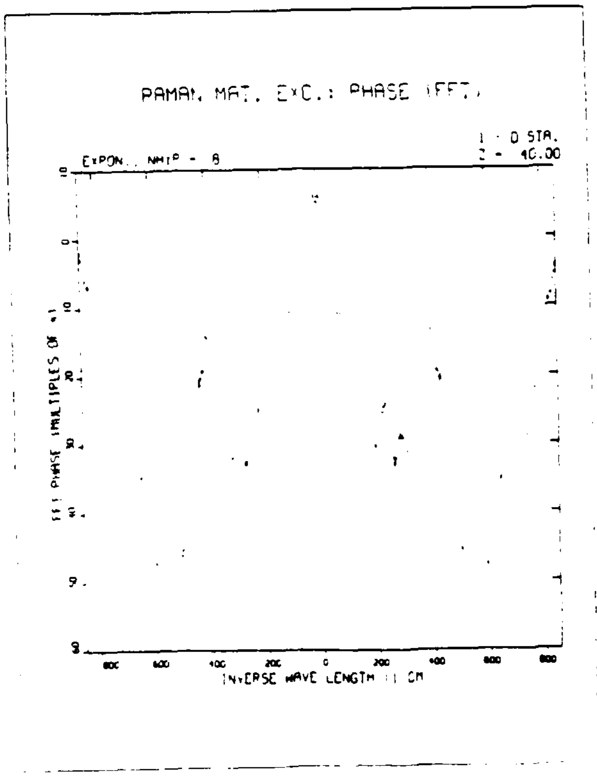


PLT2.DAT (Example B1)







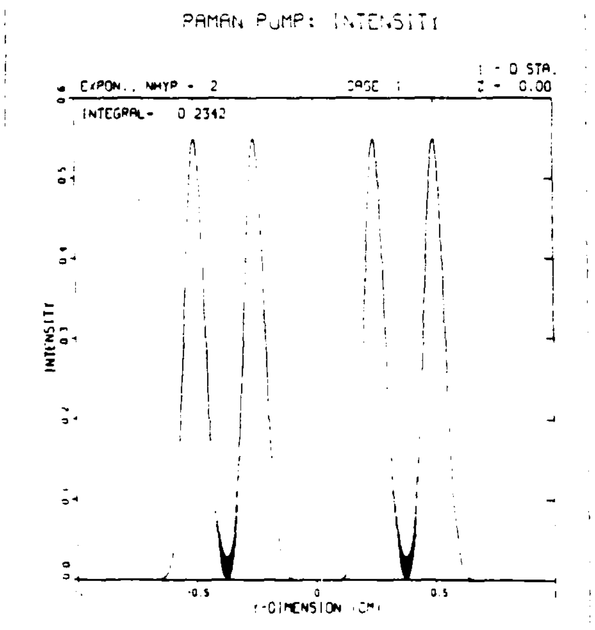


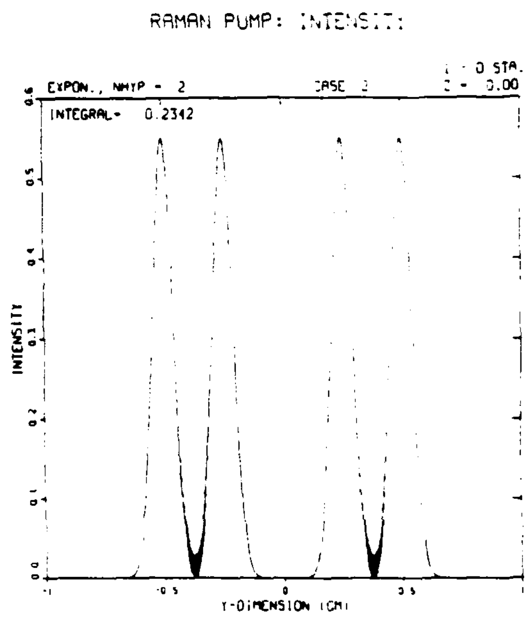
EXAMPLE B2

PLT2.DAT (Example B2)

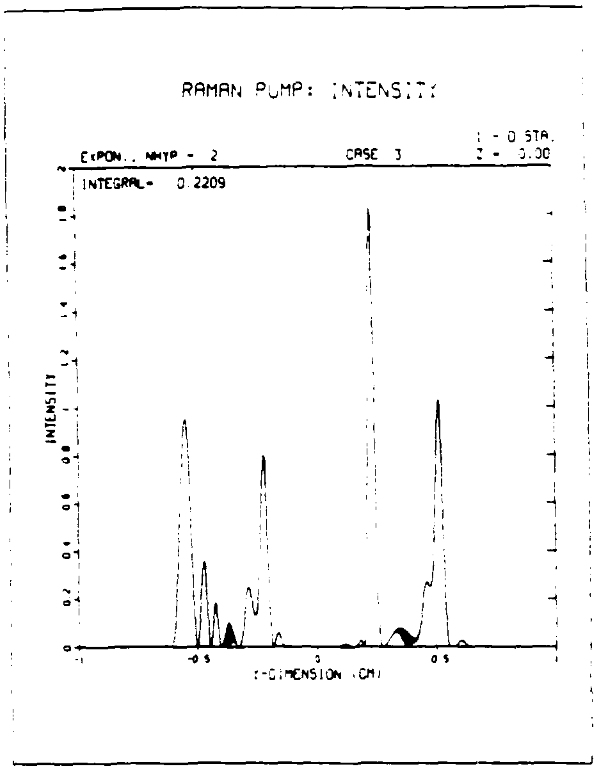
```
                 LIST OF INPUT PARAMETERS

ICONO     =      4
NHYP      =      2
NMAX      =      4000
NPUMP     =      4
NT        =      3
NY        =      4096
GAIN      =       0.4000
RIST      =       1.00*10^12
RKP       =       1.18*10^5
RKS       =       9.19*10^4
TTWO      =       633.00
YOST      =       0.0000
YWST      =       0.1000
ZFINAL    =       40.000
ZINT      =       20.000
ZKEEP     =       10.000
ZSTEP     =       0.0500
```

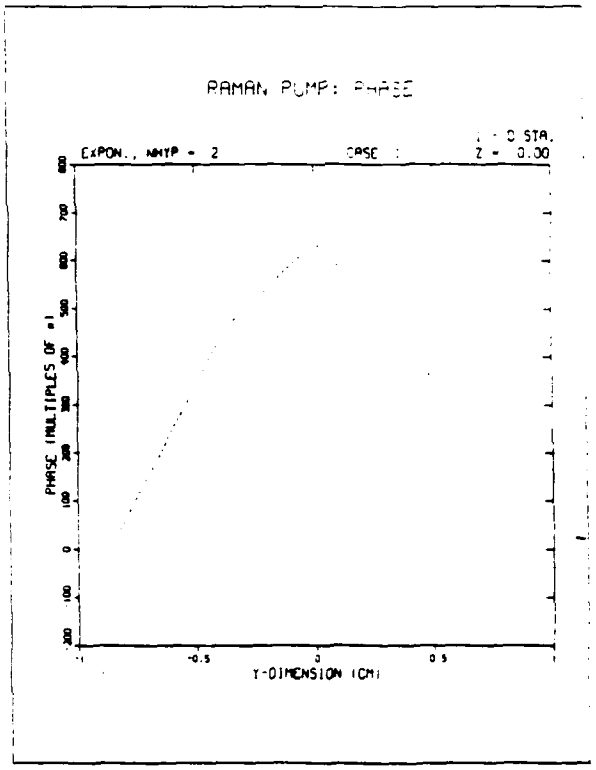
```
LIST OF INPUT PARAMETERS  CONTD

RABAMP(1-8)=      0.0000    0.0000    0.0000    0.0000    0.0000
                  0.0000    0.0000    0.0000
ROSLIM(1-8)=      0.0000    2.0000    2.0000    1.0000    1.0000
                  1.0000    1.0000    1.0000
RINT(1-10) =      0.5500    0.5500    0.5500    0.5500    0.5500
                  0.5500    0.5500    0.5500    0.5500    0.5500
YOFF(1-10) =     -0.5000   -0.2500    0.2500    0.5000    0.0000
                  0.0000    0.0000    0.0000    0.0000    0.0000
YM(1,2)    =     -1.0000    1.0000
YWIDTH     =      0.1000    0.1000    0.1000    0.1000    0.1000
                  0.1000    0.1000    0.1000    0.1000    0.1000
```

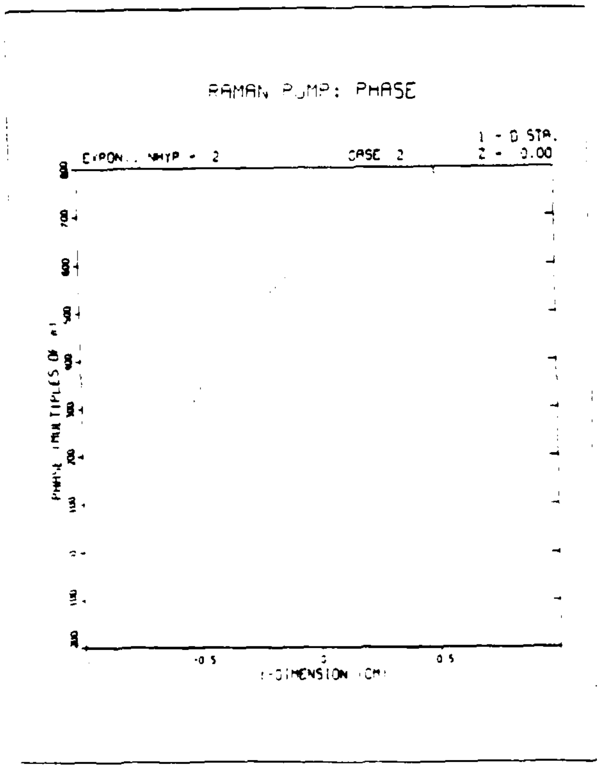


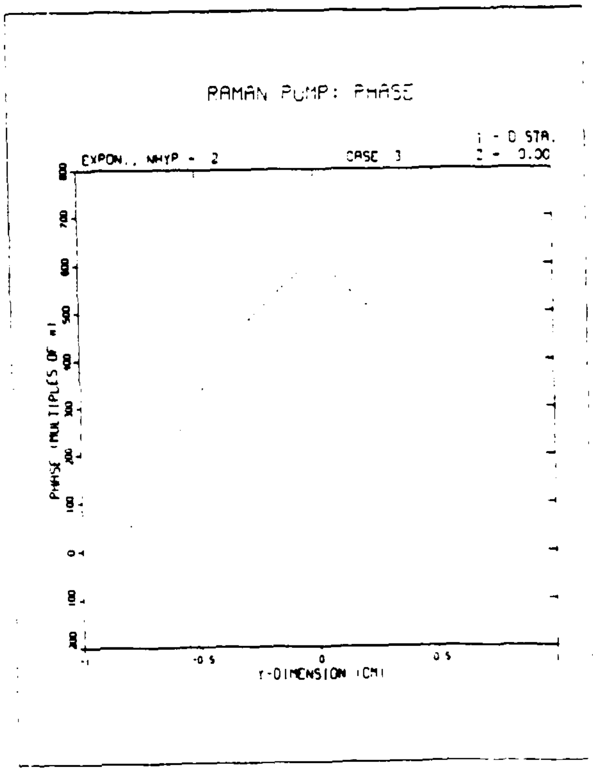


PLT2.DAT (Example B2)

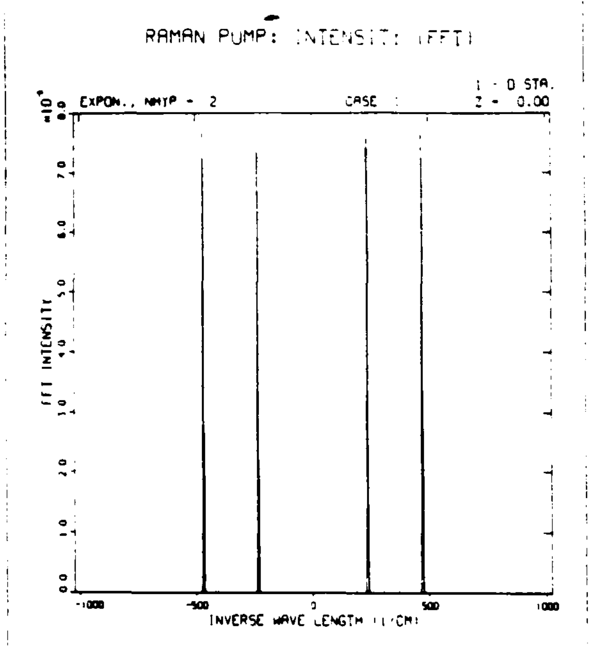


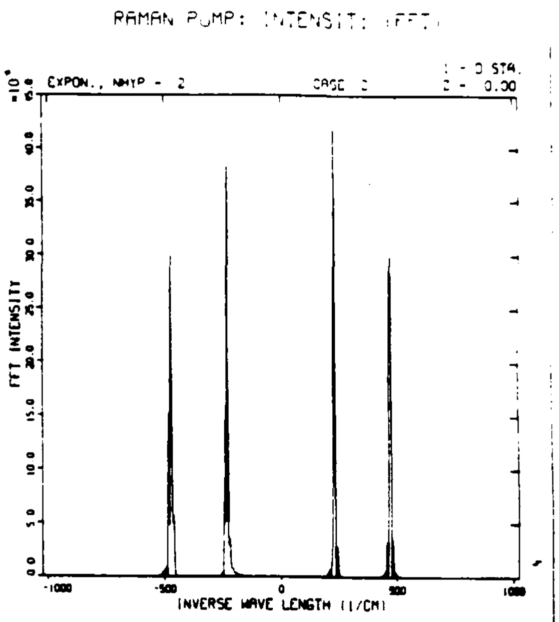


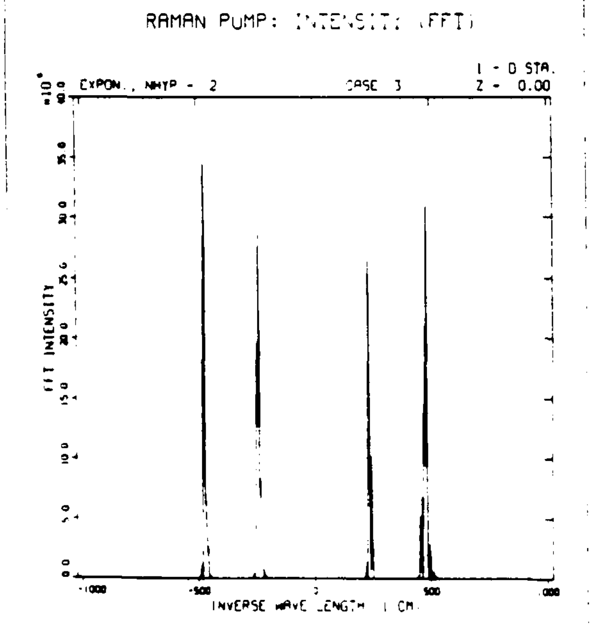


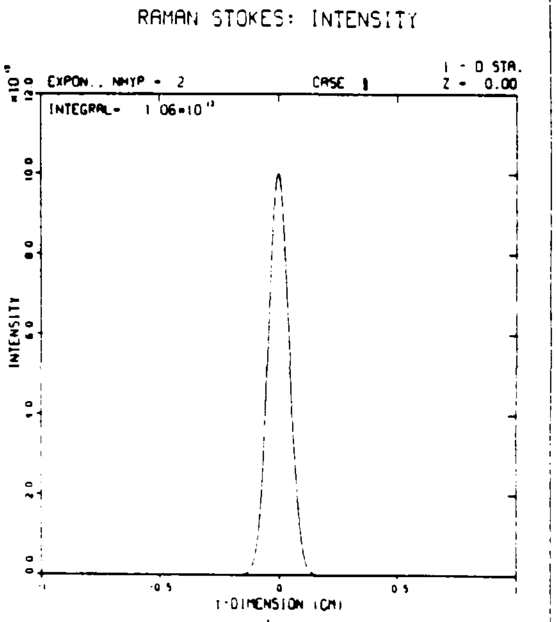


PLT2.DAT (Example B2)

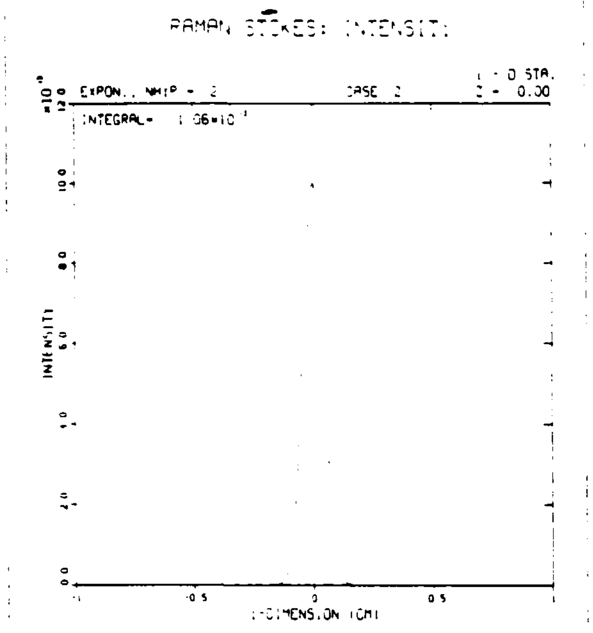


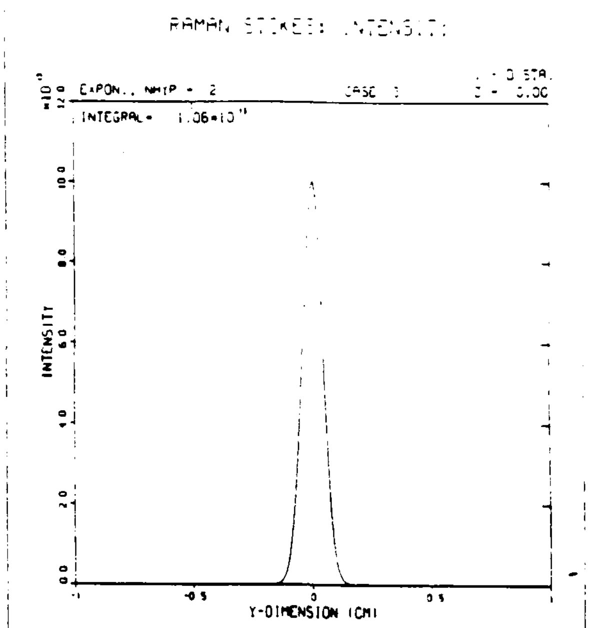


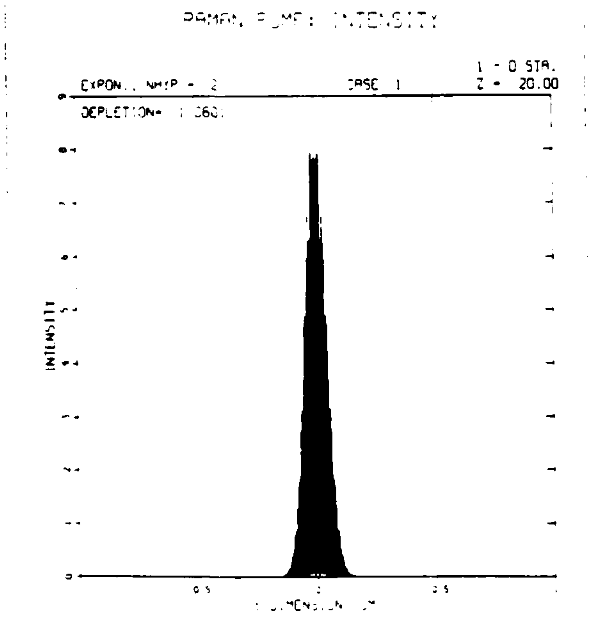


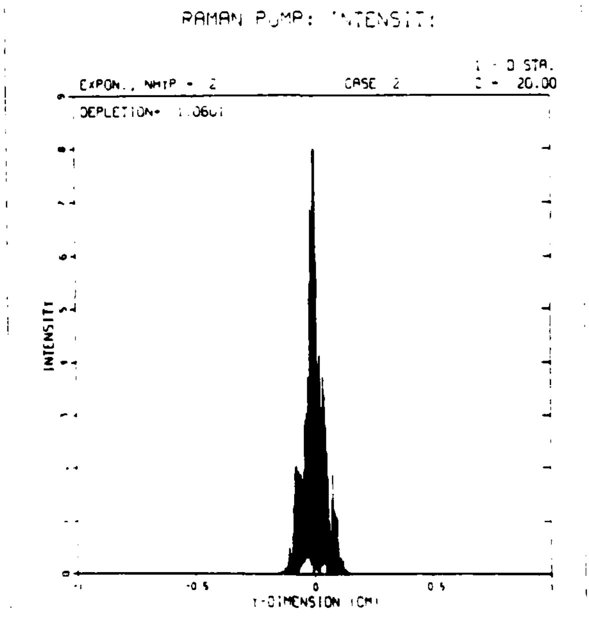


PLT2.DAT (Example B2)

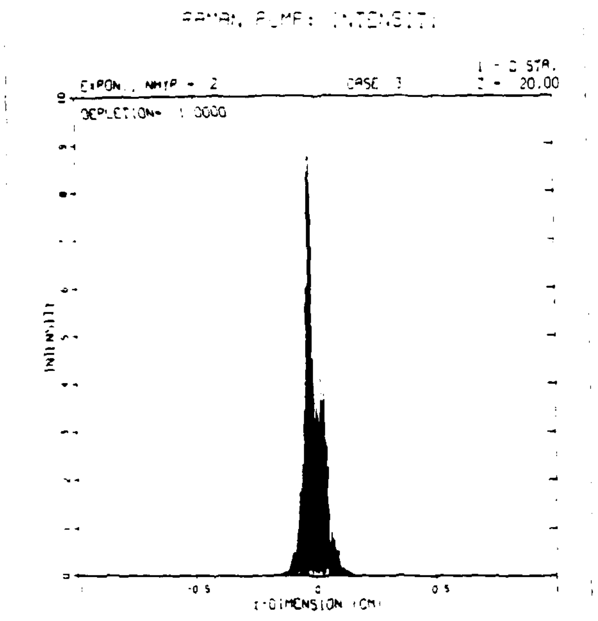


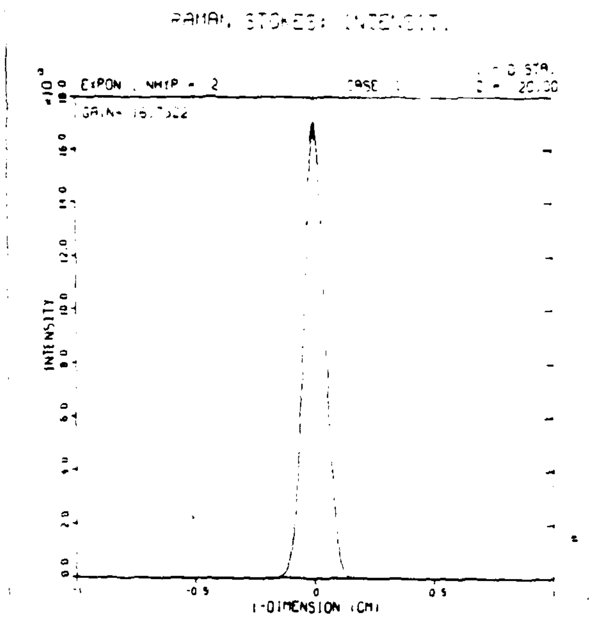


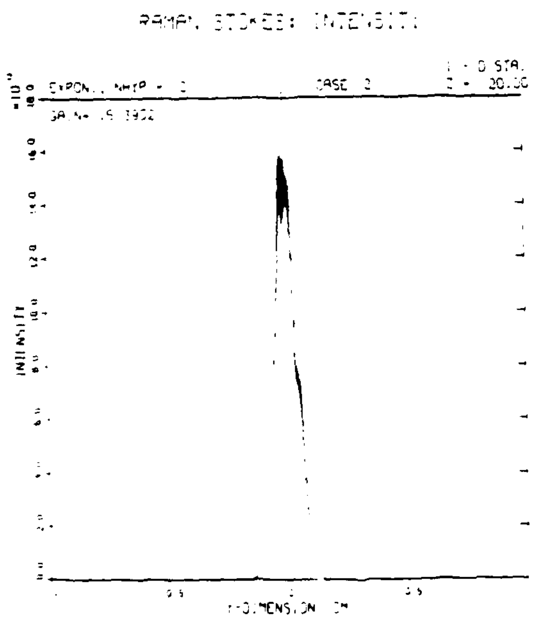


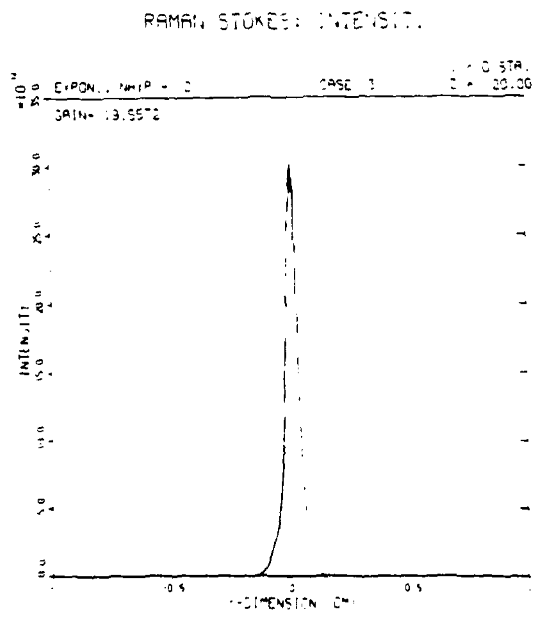


PLT2.DAT (Example B2)

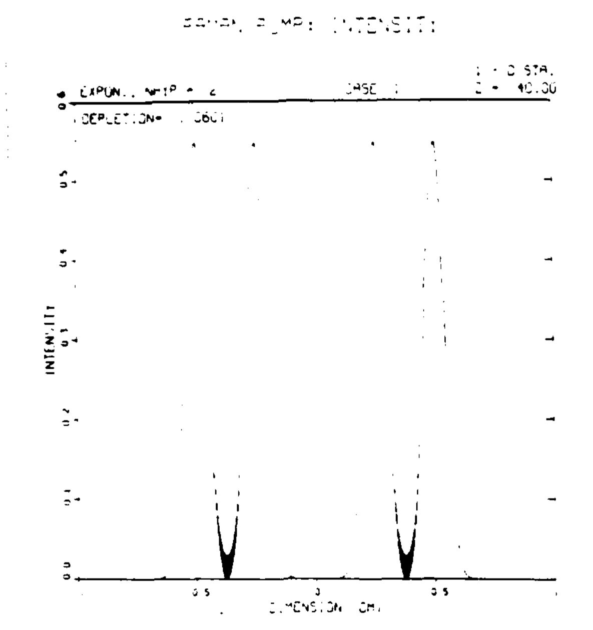


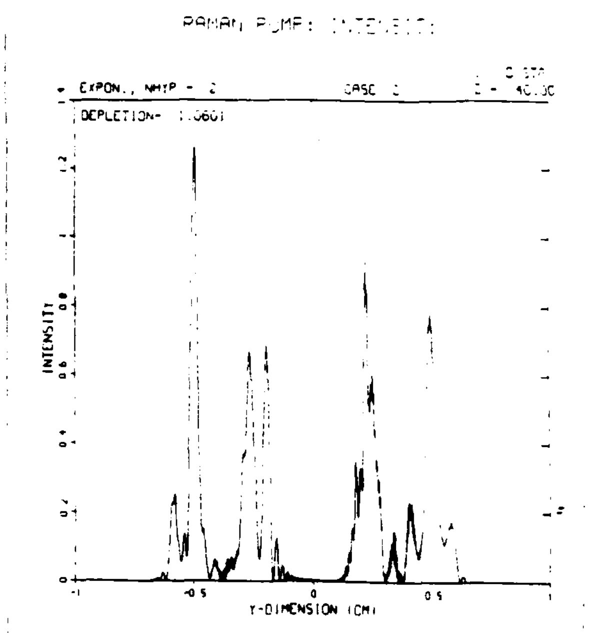


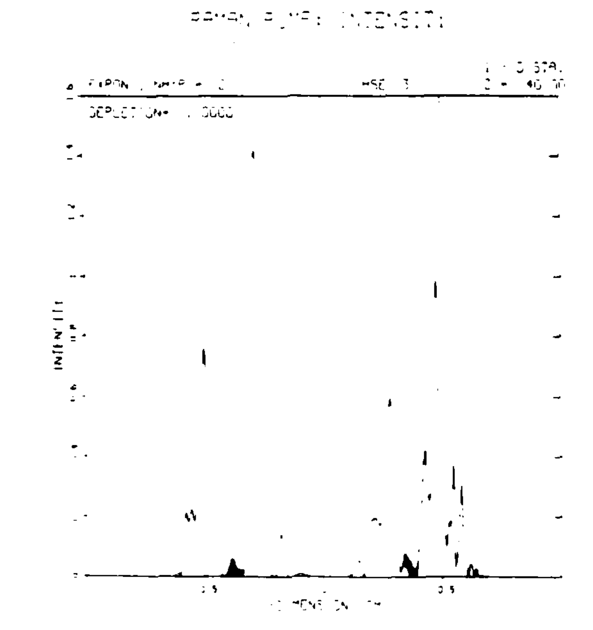


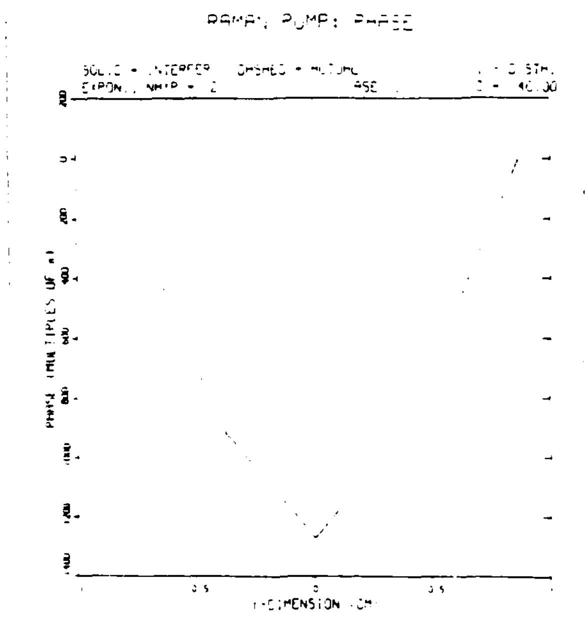


PLT2.DAT (Example B2)

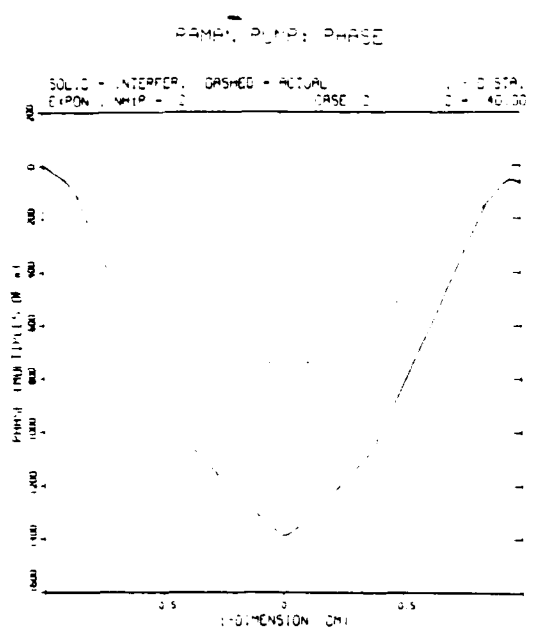


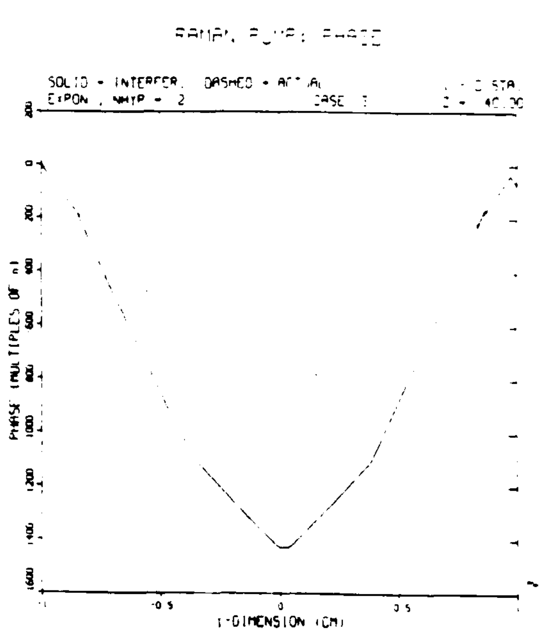


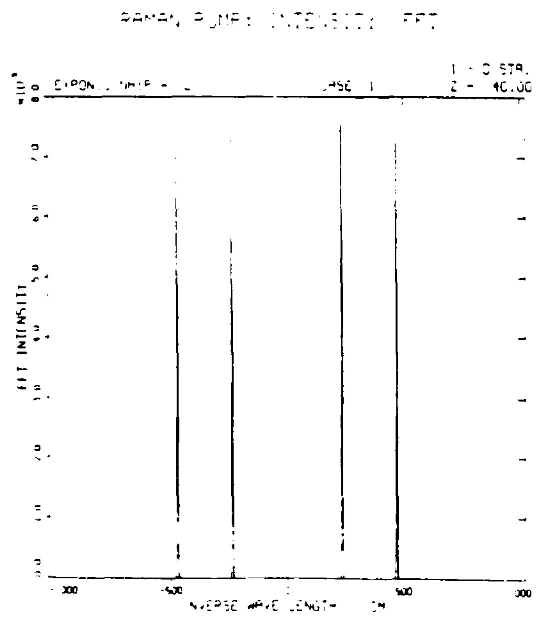


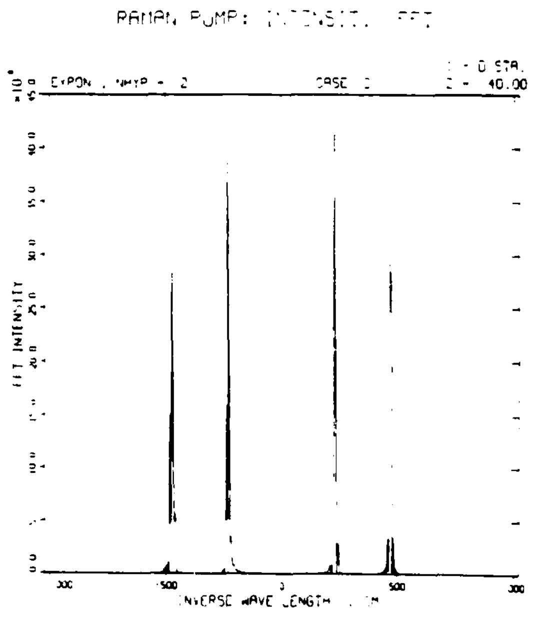


PLT2.DAT (Example B2)

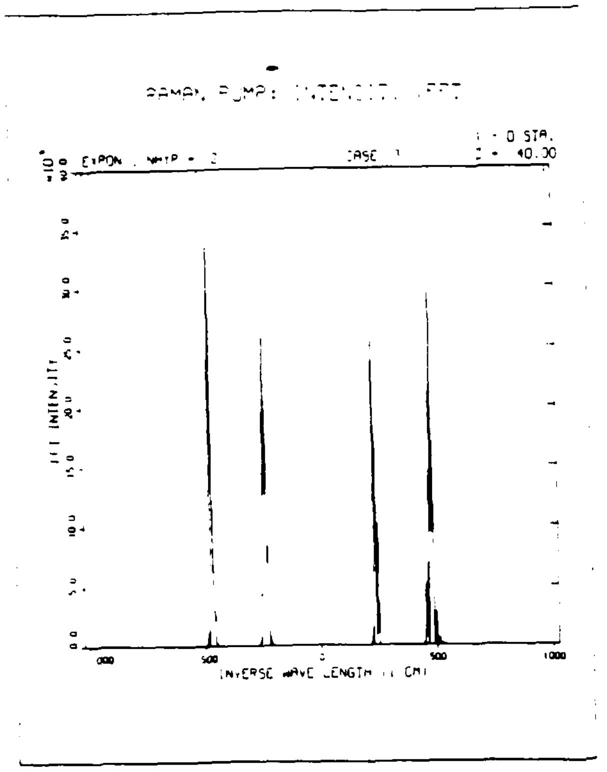


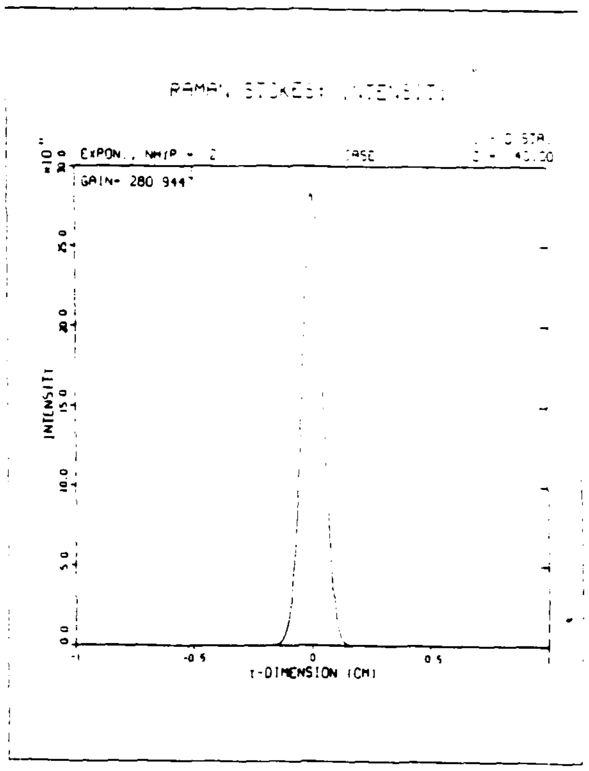


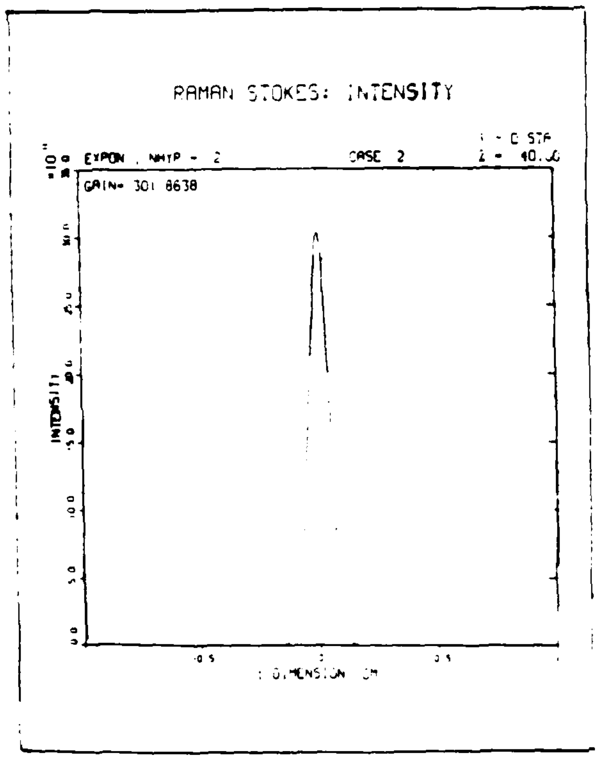


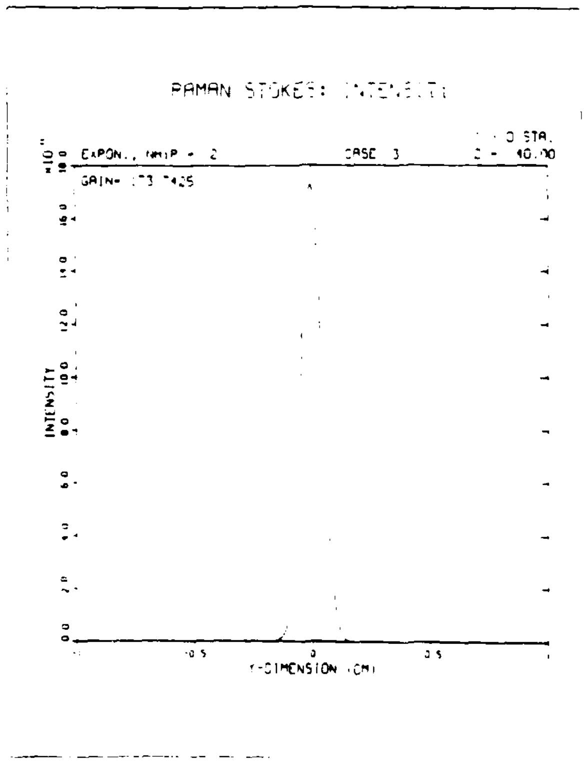


PLT2.DAT (Example B2)

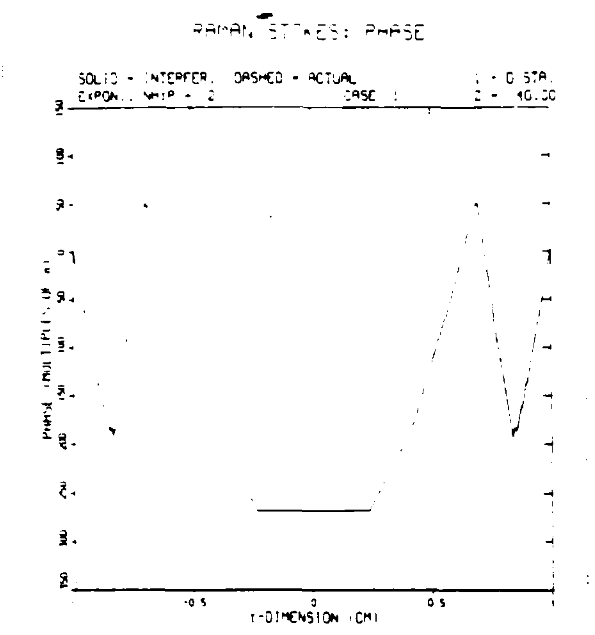


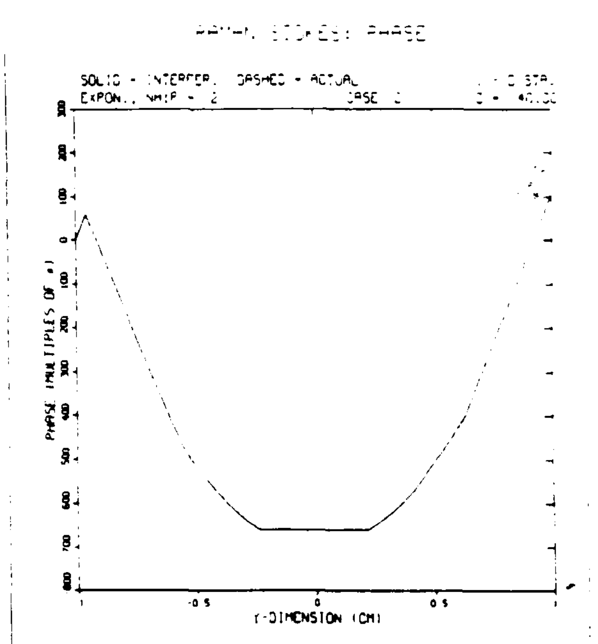


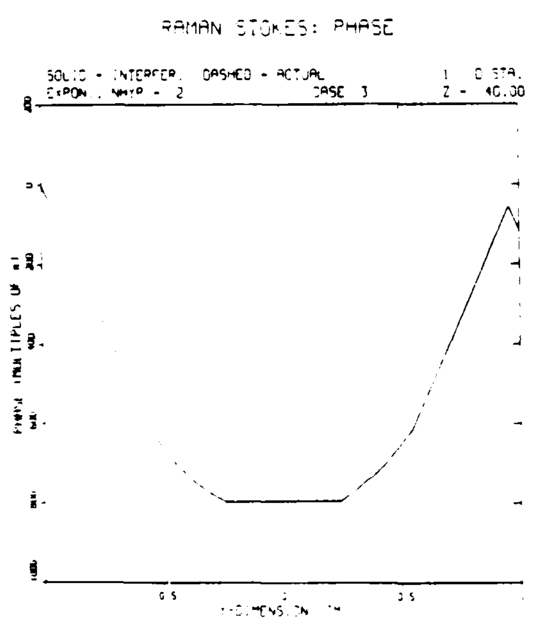


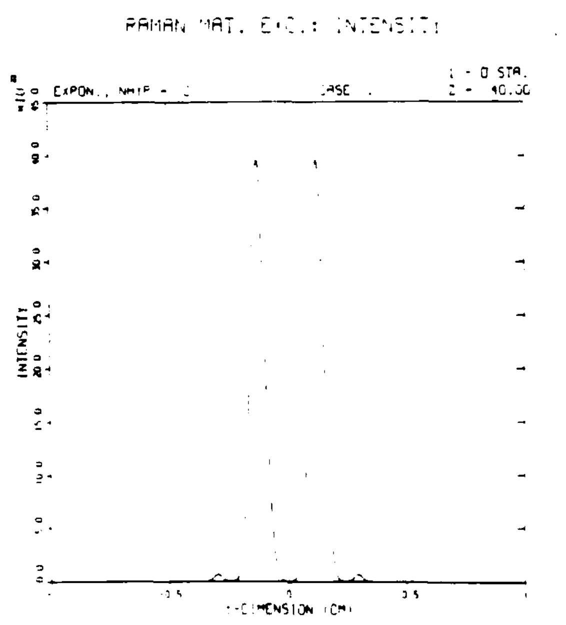


PLT2.DAT (Example B2)

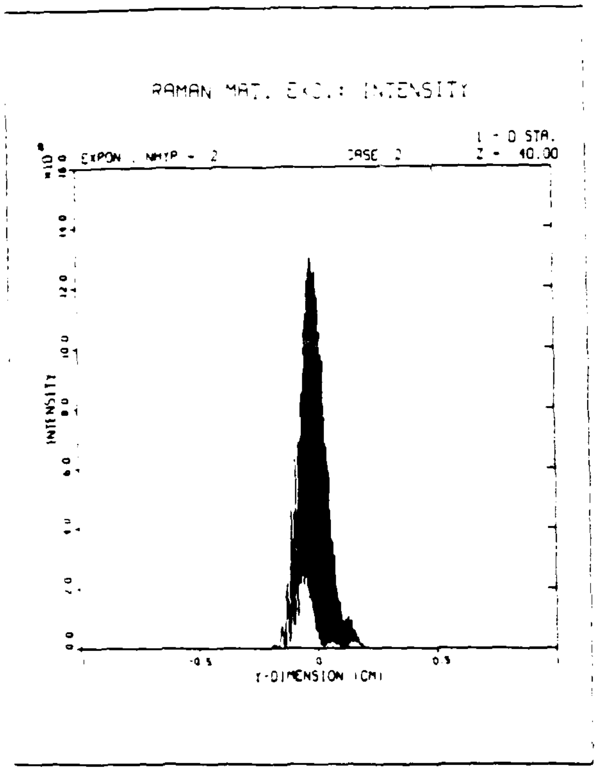


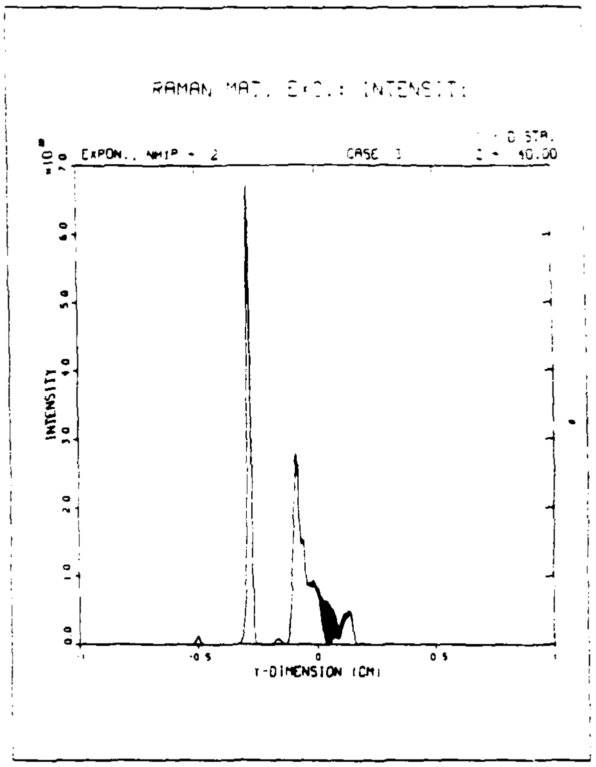






PLT2.DAT (Example B2)





XRL3.CPR

```
09:07:40 5855    0 0000   CSP    ...........................................................................
09:07:40 5858    0 0000   CSP    *                                                                         *
09:07:40 5861    0 0000   CSP    *                       WELCOME TO THE NRL CRAY XMP                       *
09:07:40 5863    0 0000   CSP    *                                                                         *
09:07:40 5869    0 0001   CSP    ...........................................................................
09:07:40 5871    0 0001   CSP    * The CRAY will be unavailable Sunday April 24 from 8:00 A M  to 4:00 P M *
09:07:40 5874    0 0001   CSP    * for software testing                                                    *
09:07:40 5878    0 0001   CSP    ***************************************************************************
09:07:40 5880    0 0001   CSP    *  There will be no CRAY off line data set recalls on Tuesday or Wednesday *
09:07:40 5884    0 0001   CSP    *  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP *
09:07:40 5887    0 0001   CSP    *  runs on our CRAY archive tape library                                  *
09:07:40 5889    0 0002   CSP    ***************************************************************************
09:07:40 5915    0 0002   CSP           CRAY X MP SERIAL-415 65     NAVAL RESEARCH LABORATORY     04 21 88
09:07:40 5918    0 0002   CSP
09:07:40 5921    0 0002   CSP           CRAY OPERATING SYSTEM          COS 1 15  ASSEMBLY DATE 01 04 88
09:07:40 5924    0 0002   CSP
09:07:40 5927    0 0002   CSP
09:07:40 6021    0 0002   CSP     JOB.JN-XR3L.MFL-511000 US-DEFER
09:07:40 6376    0 0012   CSP     ACCOUNT.AC= US= UPW= APW=
09:07:41 7479    0 1113   USER   AC213 - '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
09:07:42 0314    0 1144   USER   AUDIT
09:07:54 9445    0 3470   USER   AU003 -       214 DATASETS.    226297 BLOCKS.    115795201 WORDS
09:07:54 9449    0 3471   USER   AU003 -        64 DATASETS.     46406 BLOCKS.     23744066 WORDS ONLINE
09:07:54 9454    0 3472   USER   AU003         150 DATASETS.    179891 BLOCKS.     92051135 WORDS OFFLINE
09:07:54 9521    0 3473   CSP    FETCH.   DN-NRAM.TEXT- 'R3L DAT'
09:07:56 3078    0 3474   SCP     VAX TO CRAY: %SYSTEM - NORMAL. normal successful completion
09:07:56 3081    0 3474   SCP     VAX TO CRAY: FILE=$1$DUA.07:[HILFER.FR2]NR3L.DAT:4
09:07:56 3084    0 3474   SCP     VAX TO CRAY: 808 BYTES TRANSFERRED
09:08:00 5750    0 3474   SCP    SS004 - DATASET RECEIVED FROM FRONT END
09:08:00 7286    0 3476   CSP    ACCESS.  DN-XR3L
09:08:00 9769    0 3476   PDM    PD000 - PDN = XR3L               ID =             ED =    1  OWN = HILFER
09:08:00 9772    0 3476   PDM    PD000 - ACCESS  COMPLETE
09:08:01 0343    0 3478   CSP    XR3L.
09:09:11 1653   60 3756   ABORT  AB023 - JOB TIME LIMIT EXCEEDED
09:09:11 1655   60 3756   ABORT  AB000 - JOB STEP ABORTED      P - 01261306b
09:09:11 1658   60 3756   ABORT  AB000 - BASE     13661000 LIMIT  15225000 CPU NUMBER   01
09:09:11 1661   60 3756   ABORT  TB001 - BEGINNING OF TRACEBACK
09:09:11 1663   60 3756   ABORT        - CFFT2                           WAS CALLED BY
09:09:11 1666   60 3756   ABORT  (WCB) - CFOUR2                            AT     1206076b (LINE       16)
09:09:11 1669   60 3756   ABORT  (WCB) - DERIV                             AT     1151416b (LINE       12)
09:09:11 1674   60 3756   ABORT  (WCB)   RAM2D1C                           AT     1053354c (LINE      228)
09:09:11 1678   60 3756   ABORT  TB002   END OF TRACEBACK
09:09:11 1681   60 3756   EXP    EXIT
09:09:11 1706   60 3756   CSP    END OF JOB
09:09:11 1709   60 3757   CSP
09:09:11 1711   60 3757   CSP
09:09:11 3610   60 3758   USER        JOB NAME -                              XR3L
09:09:11 3614   60 3758   USER        USER NUMBER                             HILFER
09:09:11 3617   60 3758   USER        JOB SEQUENCE NUMBER -                       40343
09:09:11 3620   60 3758   USER
09:09:11 3624   60 3758   USER        TIME EXECUTING IN CPU -                 0000:01.00 3758
09:09:11 3627   60 3758   USER        TIME WAITING TO EXECUTE                 0000:00:16 1166
09:09:11 3631   60 3759   USER        TIME WAITING FOR I O -                  0000:00:13 2581
09:09:11 3634   60 3759   USER        TIME WAITING IN INPUT QUEUE -           0000:00:00 0722
09:09:11 3637   60 3759   USER        MEMORY * CPU TIME (MWDS*SEC) -                23 04746
09:09:11 3641   60 3759   USER        MEMORY * I O WAIT TIME (MWDS*SEC) -            1 84563
09:09:11 3644   60 3759   USER        MINIMUM JOB SIZE (WORDS) -                  44544
09:09:11 3647   60 3759   USER        MAXIMUM JOB SIZE (WORDS)                   383488
09:09:11 3651   60 3760   USER        MINIMUM FL (WORDS) -                        40960
09:09:11 3654   60 3760   USER        MAXIMUM FL (WORDS) -                       378880
```

XRL3.CPR

```
09:09:11.3658   60.3760   USER        MINIMUM JTA (WORDS) -                         3584
09:09:11.3661   60.3760   USER        MAXIMUM JTA (WORDS) -                         4608
09:09:11.3664   60.3760   USER        DISK SECTORS MOVED                            2770
09:09:11.3667   60.3760   USER        FSS SECTORS MOVED -                              0
09:09:11.3671   60.3760   USER        USER I/O REQUESTS -                            747
09:09:11.3674   60.3760   USER        USER I/O SUSPENSIONS -                        1163
09:09:11.3677   60.3760   USER        OPEN CALLS -                                    20
09:09:11.3681   60.3760   USER        CLOSE CALLS -                                   18
09:09:11.3684   60.3760   USER        MEMORY RESIDENT DATASETS -                       0
09:09:11.3687   60.3760   USER        TEMPORARY DATASET SECTORS USED -                 0
09:09:11.3691   60.3761   USER        PERMANENT DATASET SECTORS ACCESSED -          1594
09:09:11.3694   60.3761   USER        PERMANENT DATASET SECTORS SAVED -                0
09:09:11.3697   60.3761   USER        SECTORS RECEIVED FROM FRONT END -                1
09:09:11.3701   60.3761   USER        SECTORS QUEUED TO FRONT END -                    0
09:09:11.6604   60.3837   USER
09:09:11.6608   60.3837   USER    ..........................................................................
09:09:11.6610   60.3838   USER                      ***   COST TABLE FOR THIS JOB ***
09:09:11.6613   60.3839   USER                JOBNAME  -----------                             XR3L
09:09:11.6617   60.3840   USER                USER IDENT.---------                             HILFER
09:09:11.6620   60.3841   USER                BEGAN EXECUTION ----  THU APR 21, 1988    09:07.40  HOURS
09:09:11.6624   60.3842   USER                AT A PRIORITY OF  --                            3
09:09:11.6628   60.3843   USER                AND JOB CLASS OF  --                       DSMALL
09:09:11.6717   60.3844   USER     60.382510  SECONDS OF CPU TIME      @ $ 630.00  HR    --   $     10.57
09:09:11.6721   60.3845   USER     23.047894  MEMORY*CPU (MWRD-SEC)    @ $  84.00  HR    --   $      0.54
09:09:11.6725   60.3846   USER      1.847957  MEMORY*I/O (MWRD-SEC)    @ $  84.00  HR    --   $      0.04
09:09:11.6728   60.3848   USER      0.002771  I/O MEGASECTORS MOVED    @ $  84.00  EA    --   $      0.23
09:09:11.6741   60.3849   USER      0.000000  TAPE MOUNT(S)            @ $   5.00  EA         $      0.00
09:09:11.6745   60.3850   USER
09:09:11.6747   60.3850   USER                ****    TOTAL COST FOR THIS JOB     ****   --   $     11.38
09:09:11.6750   60.3850   USER    ..........................................................................
```

XPL3.CPR

```
09:10:53.3247    0.0000   CSP    ****************************************************************************
09:10:53.3250    0.0000   CSP    *                                                                          *
09:10:53.3253    0.0000   CSP    *                      WELCOME TO THE NRL CRAY XMP                         *
09:10:53.3255    0.0000   CSP    *                                                                          *
09:10:53.3258    0.0001   CSP    ****************************************************************************
09:10:53.3260    0.0001   CSP    * The CRAY will be unavailable Sunday April 24 from 8:00 A M  to 4:00 P M  *
09:10:53.3263    0.0001   CSP    * for software testing                                                     *
09:10:53.3266    0.0001   CSP    ****************************************************************************
09:10:53.3268    0.0001   CSP    *  There will be no CRAY off-line data set recalls on Tuesday or Wednesday *
09:10:53.3271    0.0001   CSP    *  mornings between 2:00 AM and 7:00 AM in order for us to perform CLEANUP *
09:10:53.3273    0.0001   CSP    *  runs on our CRAY archive tape library                                   *
09:10:53.3276    0.0001   CSP    ****************************************************************************
09:10:53.3303    0.0002   CSP            CRAY X MP SERIAL 415 65    NAVAL RESEARCH LABORATORY    04 21 88
09:10:53.3306    0.0002   CSP
09:10:53.3309    0.0002   CSP            CRAY OPERATING SYSTEM          COS 1 15  ASSEMBLY DATE 01 04 88
09:10:53.3312    0.0002   CSP
09:10:53.3314    0.0002   CSP
09:10:53.4910    0.0002   CSP     JOB.JN=XP3L.MFL=511000.US=DEFER.
09:10:53.9945    0.0013   CSP     ACCOUNT.AC=.US=.UPW=.APW=
09:10:55.0825    0.1097   USER   AC213 - '' TOTAL BUDGET WARNING LEVEL REACHED FOR THIS ACCOUNT NUMBER
09:10:55.8605    0.1127   USER   AUDIT
09:11:10.7238    0.3403   USER   AU003 -       214 DATASETS.   226297 BLOCKS.   115795201 WORDS
09:11:10.7242    0.3404   USER   AU003 -        64 DATASETS.    46406 BLOCKS.    23744066 WORDS ONLINE
09:11:10.7246    0.3405   USER   AU003 -       150 DATASETS.   179891 BLOCKS.    92051135 WORDS OFFLINE
09:11:10.7326    0.3409   CSP    ACCESS.  DN-DISLIB.ID-DISSPLA.OWN-LIBRARY.
09:11:11.0174    0.3409   PDM    PD000 - PDN - DISLIB            ID - DISSPLA   ED -    1  OWN - LIBRARY
09:11:11.0176    0.3409   PDM    PD000 - ACCESS  COMPLETE
09:11:11.0195    0.3412   CSP    ACCESS.  DN=INTLIB.ID-DISSPLA.OWN=LIBRARY.
09:11:11.2570    0.3413   PDM    PD000 - PDN - INTLIB            ID - DISSPLA   ED -    1  OWN - LIBRARY
09:11:11.2572    0.3413   PDM    PD000 - ACCESS  COMPLETE
09:11:11.2590    0.3416   CSP    ACCESS.  DN-DVSD.ID-DISSPLA.OWN-LIBRARY.
09:11:11.4937    0.3417   PDM    PD000 - PDN - DVSD              ID - DISSPLA   ED -    1  OWN - LIBRARY
09:11:11.4939    0.3417   PDM    PD000 - ACCESS  COMPLETE
09:11:11.4956    0.3418   CSP    ACCESS.  DN=XP3L.
09:11:11.7657    0.3419   PDM    PD000 - PDN = XP3L              ID =           ED =    1  OWN = HILFER
09:11:11.7659    0.3419   PDM    PD000 - ACCESS  COMPLETE
09:11:11.7675    0.3419   CSP    FETCH.   DN=NPRAM1.TEXT= NP3L.DAT
09:11:13.9169    0.3421   SCP     VAX TO CRAY: %SYSTEM S-NORMAL, normal successful completion
09:11:13.9172    0.3421   SCP     VAX TO CRAY: FILE=$1$DUA107:[HILFER FR2]NP3L.DAT:5
09:11:13.9176    0.3421   SCP     VAX TO CRAY: 1680 BYTES TRANSFERRED
09:11:18.3138    0.3421   SCP    SS004 - DATASET RECEIVED FROM FRONT END
09:11:18.5569    0.3422   CSP    XP3L.
09:11:20.1680    0.3489   PDM    PD000    PDN - F3L042188        ID -           ED -    1  OWN - HILFER
09:11:20.1683    0.3489   PDM    PD009    DATASET NOT FOUND
09:11:20.1707    0.3491   USER   I0054 - ATTEMPT TO BACKUP FROM BOD
09:11:20.1712    0.3493   USER   SL010 - READ     F3L421   READ PAST END OF DATA
09:11:20.1715    0.3493   USER   TB001 - BEGINNING OF TRACEBACK
09:11:20.1718    0.3493   USER         - $TRBK     WAS CALLED BY SLERP%    AT  1137553a
09:11:20.1721    0.3493   USER           SLERP%    WAS CALLED BY $RWDP     AT  1136510a
09:11:20.1725    0.3494   USER           $RWDP     WAS CALLED BY $RUV%     AT  1100165a
09:11:20.1728    0.3494   USER           $RUV%     WAS CALLED BY PRAM1CD   AT   102425a(LINE NUMBER      144)
09:11:20.1731    0.3494   USER   TB002 - END OF TRACEBACK
09:11:20.1736    0.3494   ABORT  AB025 - USER PROGRAM REQUESTED ABORT
09:11:20.1739    0.3494   ABORT  AB000 - JOB STEP ABORTED.    P = 01137560b
09:11:20.1741    0.3494   ABORT  AB000 - BASE    07703000 LIMIT  11226000 CPU NUMBER   00
09:11:20.1746    0.3494   EXP    EXIT.
09:11:20.1765    0.3494   CSP    END OF JOB
09:11:20.1768    0.3495   CSP
09:11:20.1770    0.3495   CSP
09:11:20.3507    0.3496   USER       JOB NAME -                        XP3L
```

XPL3.CPR

```
09:11:20.3510   0.3496   USER    USER NUMBER                              HILFER
09:11:20.3513   0.3496   USER    JOB SEQUENCE NUMBER                          40355
09:11:20.3516   0.3496   USER
09:11:20.3519   0.3496   USER    TIME EXECUTING IN CPU -                  0000:00:00.3496
09:11:20.3522   0.3496   USER    TIME WAITING TO EXECUTE -                0000:00:14.1034
09:11:20.3525   0.3496   USER    TIME WAITING FOR I/O                     0000:00:12.3816
09:11:20.3528   0.3497   USER    TIME WAITING IN INPUT QUEUE -            0000:00:00.0068
09:11:20.3532   0.3497   USER    MEMORY * CPU TIME (MWDS*SEC) -                 0.02972
09:11:20.3535   0.3497   USER    MEMORY * I/O WAIT TIME (MWDS*SEC) -            1.50167
09:11:20.3538   0.3497   USER    MINIMUM JOB SIZE (WORDS)                     44544
09:11:20.3541   0.3497   USER    MAXIMUM JOB SIZE (WORDS) -                  374784
09:11:20.3544   0.3497   USER    MINIMUM FL (WORDS) -                         40960
09:11:20.3547   0.3497   USER    MAXIMUM FL (WORDS) -                        370176
09:11:20.3550   0.3497   USER    MINIMUM JTA (WORDS)                           3584
09:11:20.3553   0.3498   USER    MAXIMUM JTA (WORDS) -                         4608
09:11:20.3556   0.3498   USER    DISK SECTORS MOVED -                          1888
09:11:20.3559   0.3498   USER    FSS SECTORS MOVED -                              0
09:11:20.3562   0.3498   USER    USER I/O REQUESTS -                            733
09:11:20.3566   0.3498   USER    USER I/O SUSPENSIONS                           958
09:11:20.3569   0.3498   USER    OPEN CALLS                                      20
09:11:20.3572   0.3498   USER    CLOSE CALLS -                                   18
09:11:20.3575   0.3498   USER    MEMORY RESIDENT DATASETS -                       0
09:11:20.3580   0.3498   USER    TEMPORARY DATASET SECTORS USED -                 0
09:11:20.3639   0.3498   USER    PERMANENT DATASET SECTORS ACCESSED -          2482
09:11:20.3642   0.3498   USER    PERMANENT DATASET SECTORS SAVED -                0
09:11:20.3645   0.3498   USER    SECTORS RECEIVED FROM FRONT END -                1
09:11:20.3648   0.3498   USER    SECTORS QUEUED TO FRONT END -                    0
09:11:20.6494   0.3572   USER
09:11:20.6496   0.3572   USER   ..........................................................................
09:11:20.6500   0.3573   USER                         ***   COST TABLE FOR THIS JOB ***
09:11:20.6503   0.3574   USER                  JOBNAME -----------                       XP3L
09:11:20.6507   0.3575   USER                  USER IDENT -------                        HILFER
09:11:20.6510   0.3576   USER                  BEGAN EXECUTION ----  THU APR 21, 1988    09:10:52  HOURS
09:11:20.6514   0.3577   USER                  AT A PRIORITY OF  --                            3
09:11:20.6517   0.3578   USER                  AND JOB CLASS OF  --                      DSMALL
09:11:20.6521   0.3579   USER       0.355991   SECONDS OF CPU TIME     @ $ 630.00  HR         $      0.06
09:11:20.6525   0.3580   USER       0.030136   MEMORY*CPU (MWRD-SEC)   @ $  84.00  HR    --   $      0.00
09:11:20.6529   0.3581   USER       1.503896   MEMORY*I/O (MWRD-SEC)   @ $  84.00  HR         $      0.04
09:11:20.6533   0.3582   USER       0.001890   I/O MEGASECTORS MOVED   @ $  84.00  EA    --   $      0.16
09:11:20.6536   0.3584   USER       0.000000   TAPE MOUNT(S)           @ $   5.00  EA    --   $      0.00
09:11:20.6540   0.3585   USER
09:11:20.6542   0.3585   USER                  ****    TOTAL COST FOR THIS JOB     ****       $      0.26
09:11:20.6545   0.3585   USER   ..........................................................................
```

```
$FLDATE
  DONYET=1,
  MONTH=03,
  DAY=30,
  YEAR=88,
  IPART=2,
  NEDN=1,
$
$CONDAT
  LPRMT(1)=1,
  LPRMT(2)=1,
  LPRMT(3)=1,
  LPRMT(4)=0,
  NSEC=3,
  CSEC(1,1)=(1.0,2.0),
  CSEC(1,2)=(2.0,2.0),
  CSEC(1,3)=(3.0,2.0),
  CSEC(2,1)=(1.0,2.0),
  CSEC(2,2)=(2.0,2.0),
  CSEC(2,3)=(3.0,2.0),
  CSEC(4,1)=(1.0,2.0),
  CSEC(4,2)=(2.0,2.0),
  CSEC(4,3)=(3.0,2.0),
  CSEC(5,1)=(1.0,2.0),
  CSEC(5,2)=(2.0,2.0),
  CSEC(5,3)=(3.0,2.0),
  CSEC(7,1)=(1.0,2.0),
  CSEC(7,2)=(2.0,2.0),
  CSEC(7,3)=(3.0,2.0),
  CSEC(8,1)=(1.0,2.0),
  CSEC(8,2)=(2.0,2.0),
  CSEC(8,3)=(3.0,2.0),
  CSEC(10,1)=(1.0,2.0),
  CSEC(10,2)=(2.0,2.0),
  CSEC(10,3)=(3.0,2.0),
  CSEC(11,1)=(1.0,2.0),
  CSEC(11,2)=(2.0,2.0),
  CSEC(11,3)=(3.0,2.0),
  CSEC(13,1)=(1.0,2.0),
  CSEC(13,2)=(2.0,2.0),
  CSEC(13,3)=(3.0,2.0),
  CSEC(14,1)=(1.0,2.0),
  CSEC(14,2)=(2.0,2.0),
  CSEC(14,3)=(3.0,2.0),
  CSEC(16,1)=(1.0,2.0),
  CSEC(16,2)=(2.0,2.0),
  CSEC(16,3)=(3.0,2.0),
  CSEC(17,1)=(1.0,2.0),
  CSEC(17,2)=(2.0,2.0),
  CSEC(17,3)=(3.0,2.0),
$
$ZPLOT
  KZ(1)=1,
  KZ(2)=2,
  KZ(3)=3,
  KZ(4)=4,
  KZ(5)=5,
$
```

NRL3.DAT

```
$NAML
  NPUMP=4,
  YM(1)=-1.0,
  YM(2)=1.0,
  YOFF(1)=-0.5,
  YOFF(2)=-0.25,
  YOFF(3)=0.25,
  YOFF(4)=0.5,
  RIST=1.0E-12,
  NHYP=2,
  RABAMP(1)=0.0,
  RABAMP(2)=0.0,
  RABAMP(3)=1.0,
  RDSLIM(1)=0.0,
  RDSLIM(2)=2.0,
  RDSLIM(3)=2.0,
  ICOND=4,
  ZFINAL=40.0,
  ZKEEP=10.0,
  GAIN=0.4,
$
     NAMELIST/NAML/NPUMP,YM,TM,ZINT,RKP,RKS,YOFF,TOFF,YWIDTH,TWIDTH,
    1 YOST,TOST,YWST,TWST,RINT,RIST,RAMASM,RALASM,NHYP,PHL,PHST,TOC,
    2 ITYPE,RTYPE,RABAMP,RDSLIM,ICOND,ZSTEP,ZFINAL,ZKEEP,NMAX,TTWO,GAIN
```

XRL3.JOB

```
AUDIT.
FETCH,    DN=NRAM,TEXT='NR3L.DAT'.
ACCESS,   DN=XR3L.
XR3L.
DISPOSE, DN=ERRM,DF=BB,WAIT,TEXT='XR3L.MSG.'.
AUDIT.
EXIT.
```

XPL3.JOB

```
AUDIT.
ACCESS,   DN=DISLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=INTLIB,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=DVSD,ID=DISSPLA,OWN=LIBRARY.
ACCESS,   DN=XP3L.
FETCH,    DN=NPRAM1,TEXT='NP3L.DAT'.
XP3L.
AUDIT.
DISPOSE,  DN=META,DF=BB,WAIT,TEXT='PLT2.DAT'.
DISPOSE,  DN=EPRM,DF=BB,WAIT,TEXT='XP3L.MSG.'.
DISPOSE,  DN=DISOUT,DF=BB,WAIT,TEXT='XP3L.DSP.'.
EXIT.
```

APPENDIX C 2-D Operation; Examples

One example is presented to illustrate the output of the codes RAM2D1 and PRAM1 in two-dimensional simulations.

EXAMPLE C

PLT2.DAT (Example C)

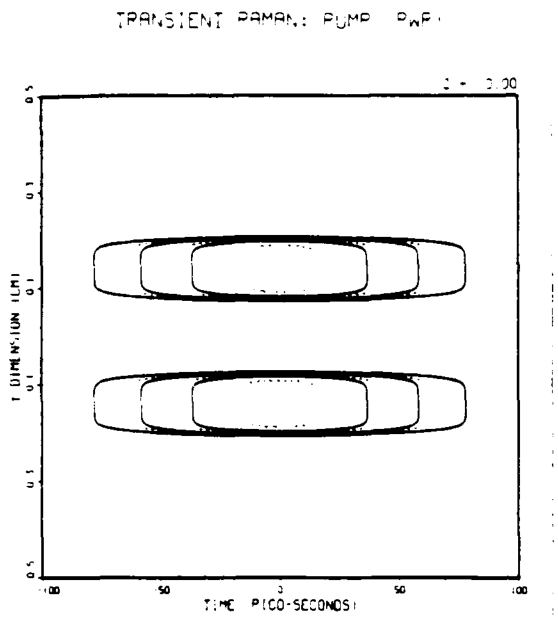
```
                LIST OF INPUT PARAMETERS

ICOND     -    2
ILN       -    1
ISHM      -    1
NDEC      -    3
NHYP      -    8
NMAX      -    4000
NPUMP     -    2
NT        -    128
NY        -    512
GAIN      -     3.0000
PHST      -     0.0000
PALASM    -     5.0000
PAMASM    -     1.5000
RIST      -     0.0002
RKP       -     1.1800*10^5
RKS       -     91893.
TDC       -     5.0000
TDST      -    -40.000
TTWO      -     633.00
TWST      -     40.000
YCST      -     0.0000
YWST      -     0.1000
ZFINAL    -     40.000
ZINT      -     20.000
ZKEEP     -     1.0000
ZSTEP     -     0.0500
```

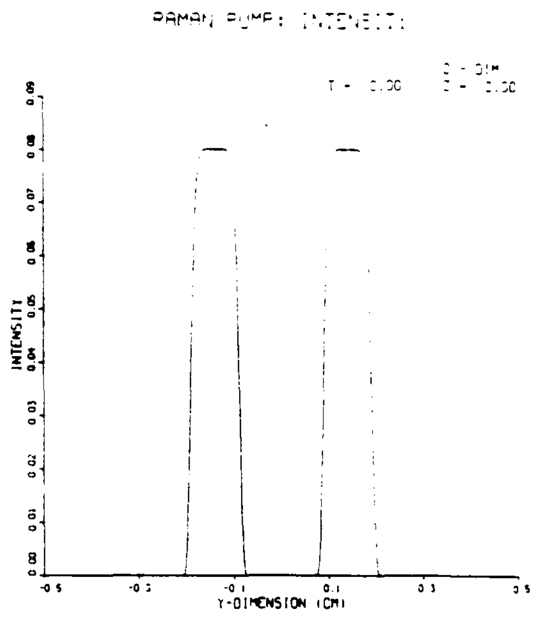
```
LIST OF INPUT PARAMETERS  CONTD.

ISRF(1-6)    -   -1     -1     -1     -1     -1     -1     -1     -1
LEVEL        -    5      3      4      5      6      7      8      9
PHL(1-10)    -    0.0000 0.0000 0.0000 0.0000 0.0000
                  0.0000 0.0000 0.0000 0.0000 0.0000
RINT(1-10)   -    0.0800 0.0800 0.5500 0.5500 0.5500
                  0.5500 0.5500 0.5500 0.5500 0.5500
TM(1,2)      -  -100.00  100.00
TOFF(1-10)   -    0.0000 0.0000 0.0000 0.0000 0.0000
                  0.0000 0.0000 0.0000 0.0000 0.0000
TWIDTH       -    40.000 40.000 40.000 40.000 40.000
                  40.000 40.000 40.000 40.000 40.000
YOFF(1-10)   -    0.1400 -0.1400 0.0000 0.0000 0.0000
                  0.0000 0.0000 0.0000 0.0000 0.0000
YM(1,2)      -   -0.5000 0.5000
YWIDTH       -    0.1000 0.1000 0.1000 0.1000 0.1000
                  0.1000 0.1000 0.1000 0.1000 0.1000
```

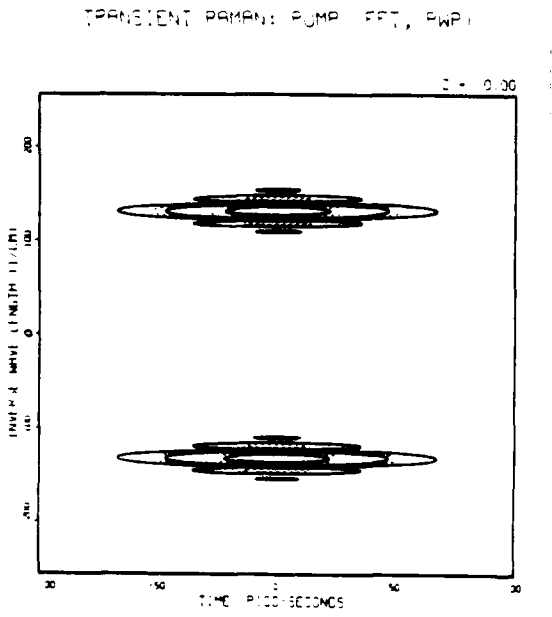
```
LIST OF INPUT PARAMETERS (CONTD)
CSEC(1-19,1-8) -
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
  0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
```

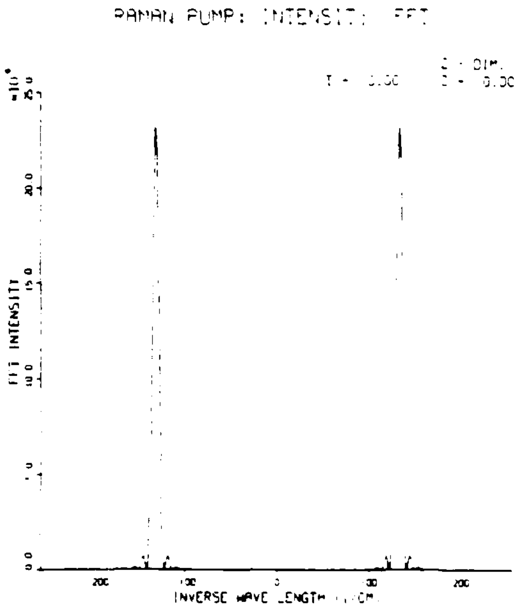
```
                LIST OF OUTPUT PARAMETERS

   PUMP       TOTAL INTENSITY       Y-WIDTH

     2           2.00*10^4            6.07
     1           2.00*10^4            6.07
```

PLT2.DAT (Example C)

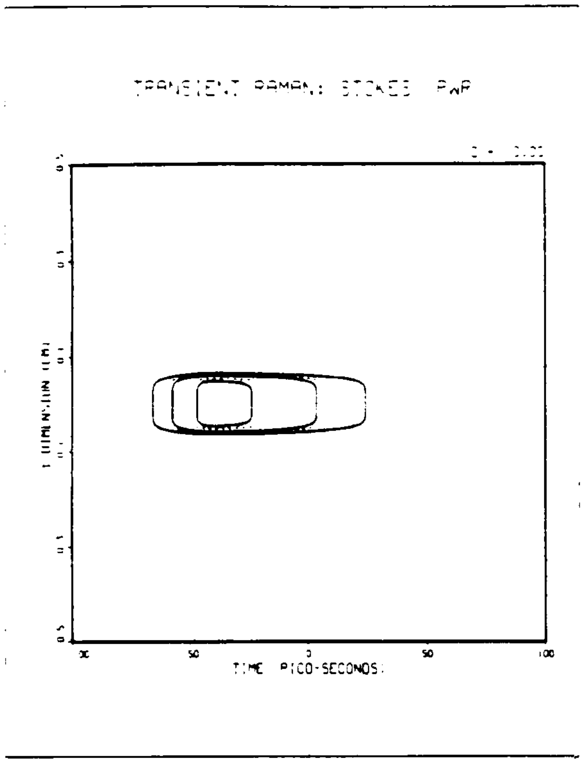


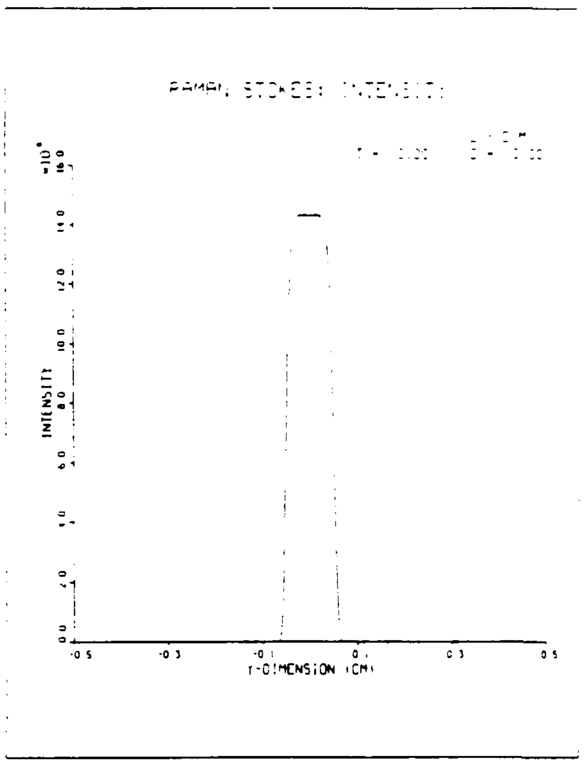


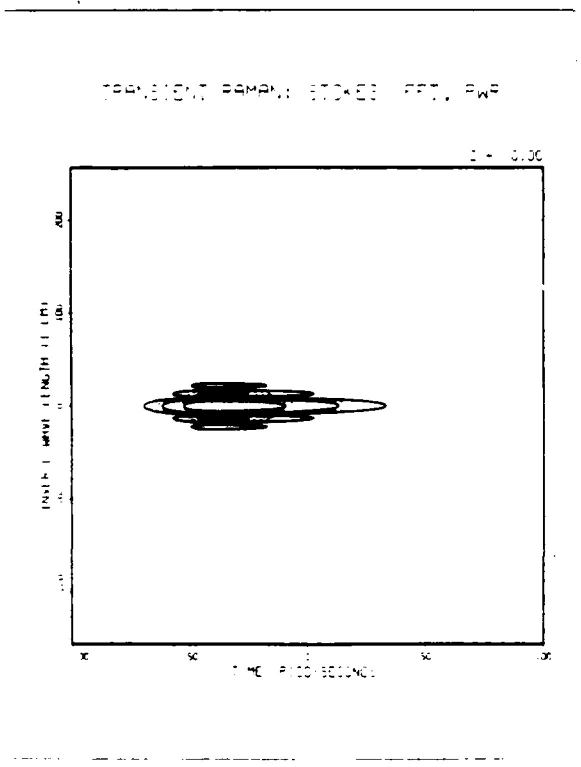


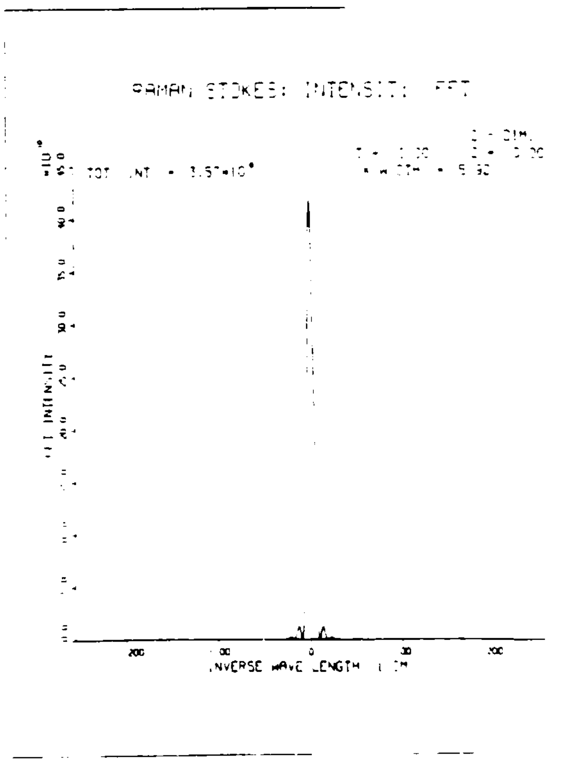


PLT2.DAT (Example C)

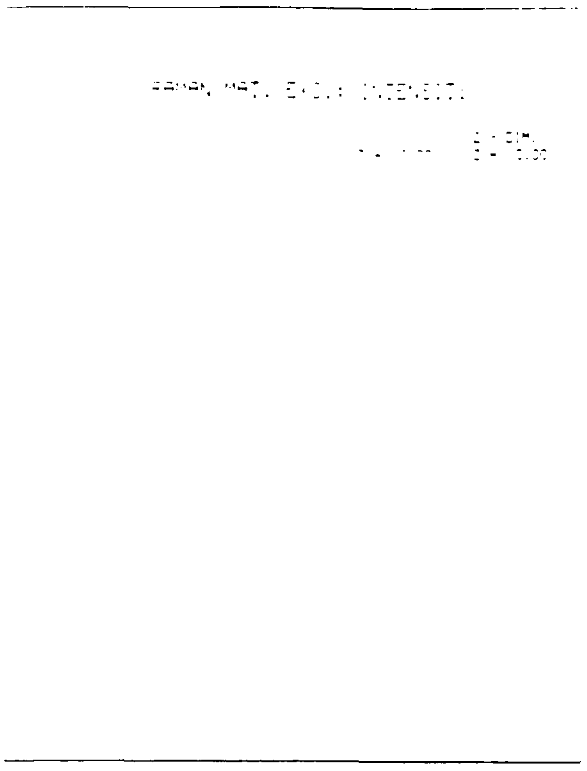


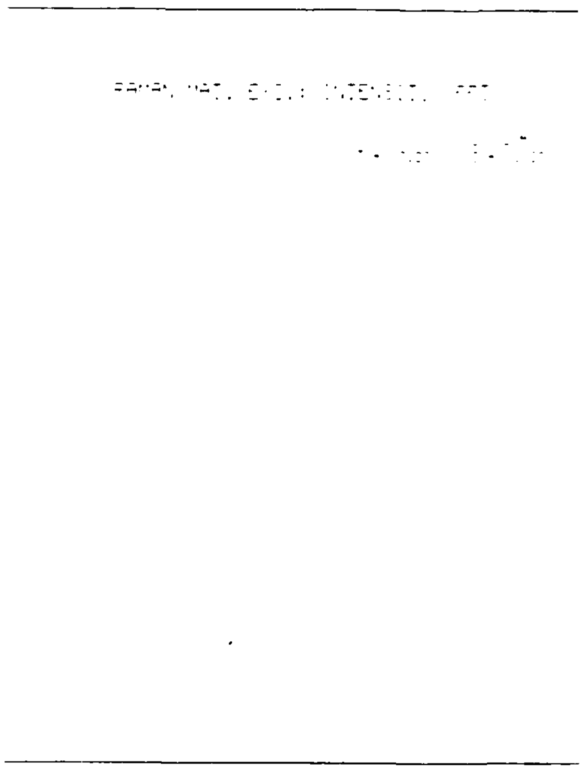


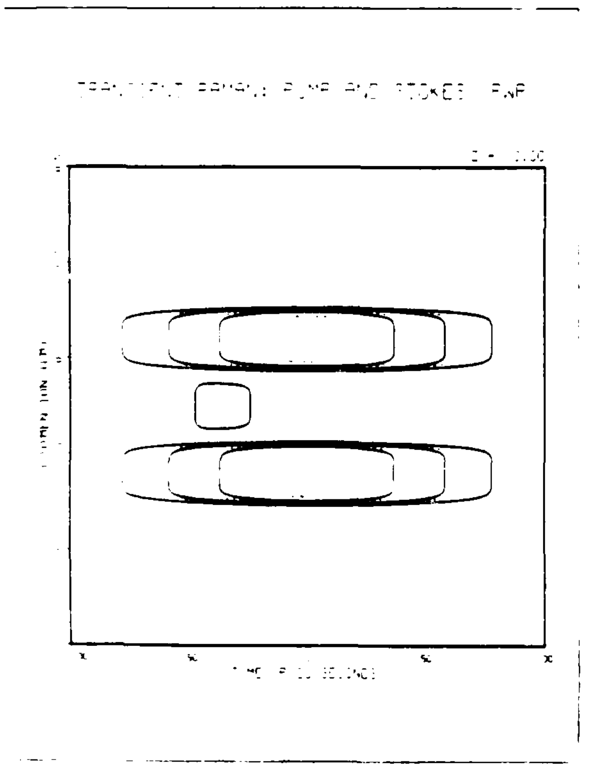


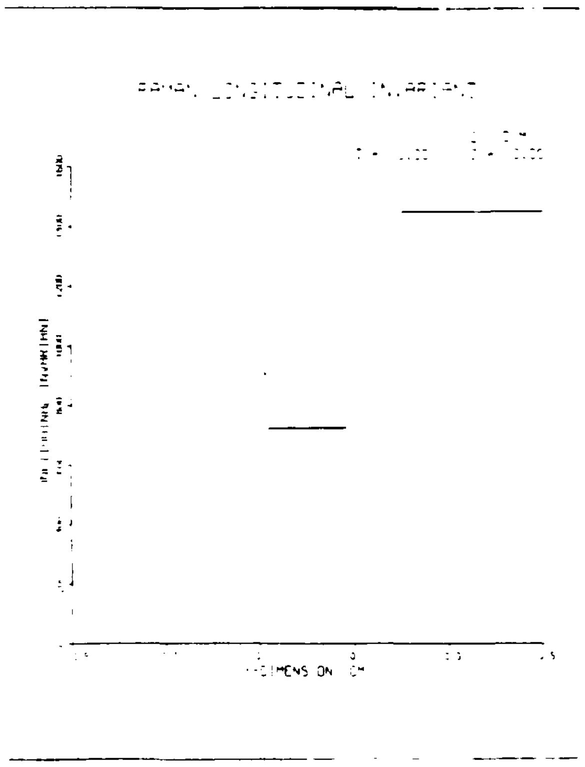


PLT2.DAT (Example C)

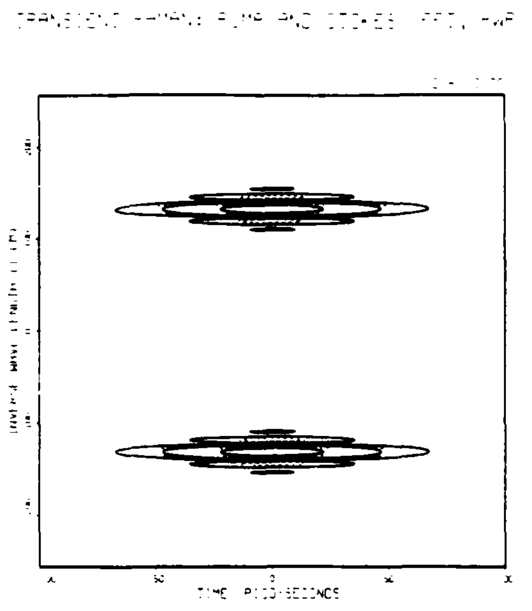


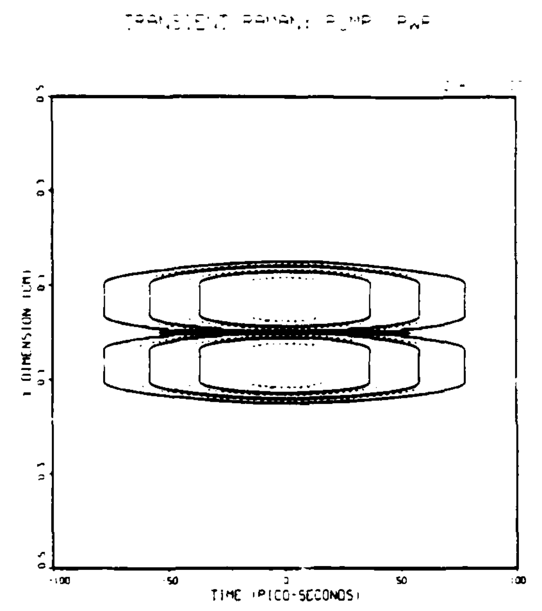


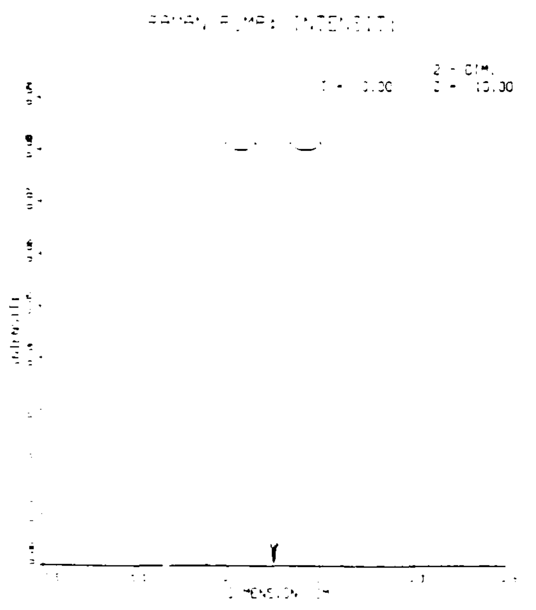


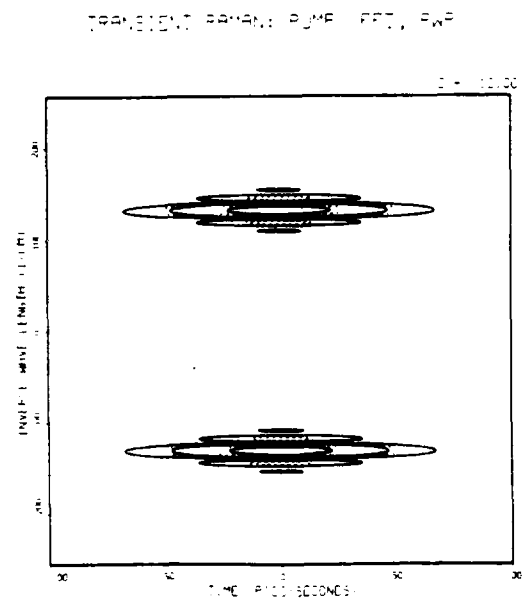


PLT2.DAT (Example C)

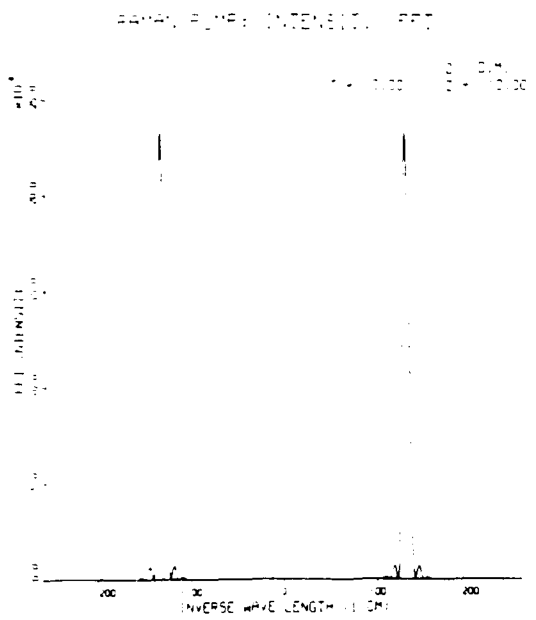


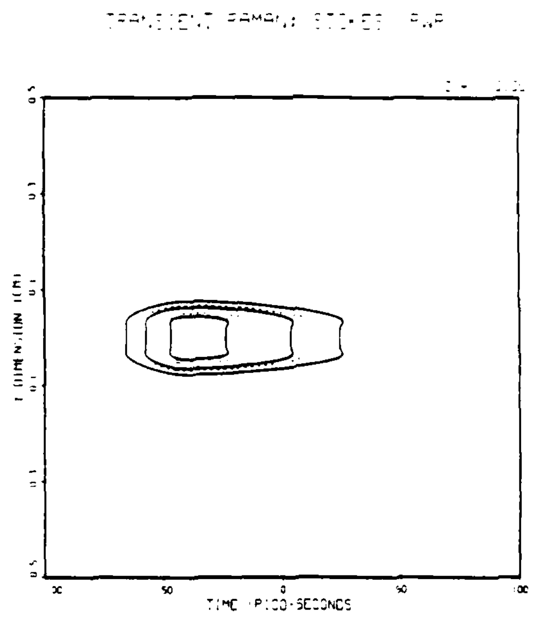


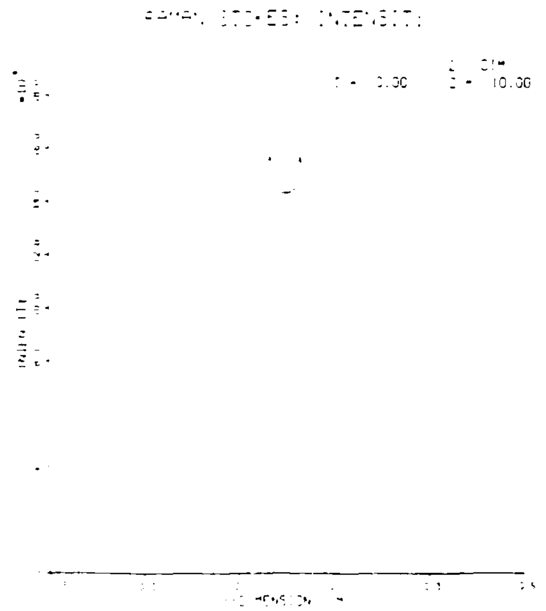


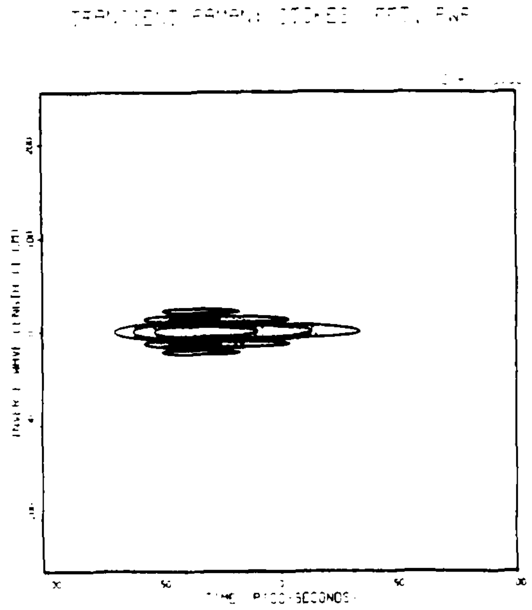


PLT2.DAT (Example C)

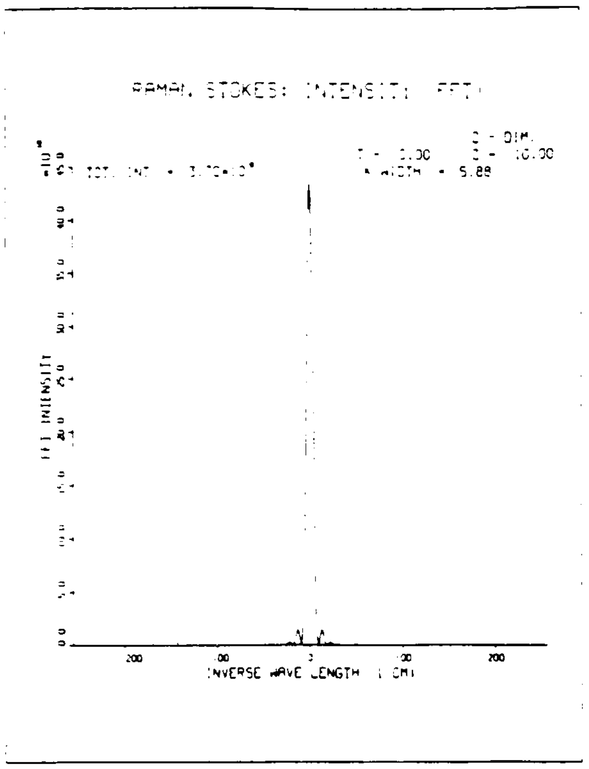


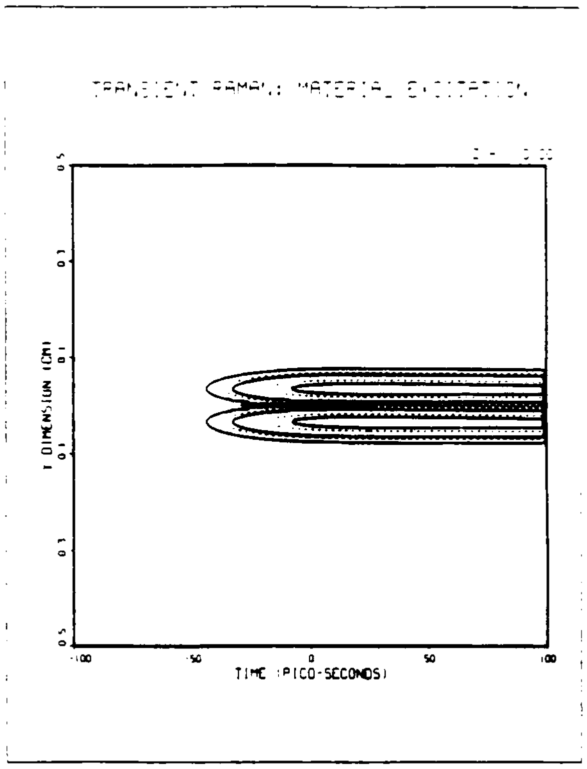


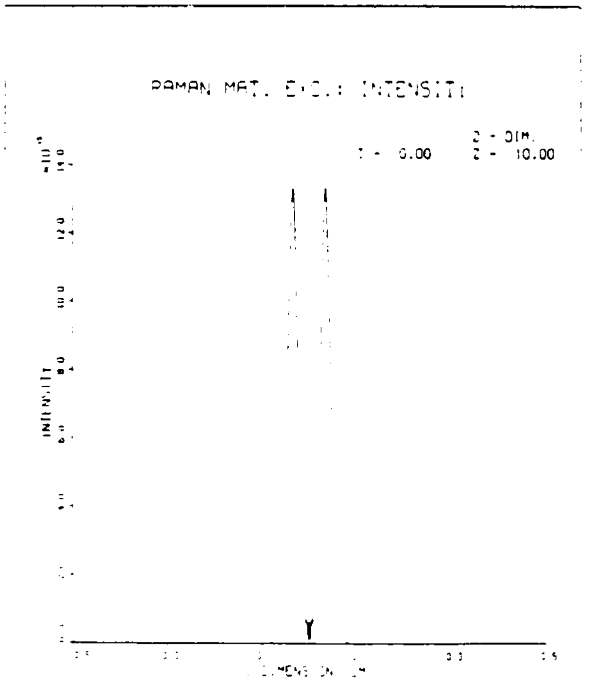


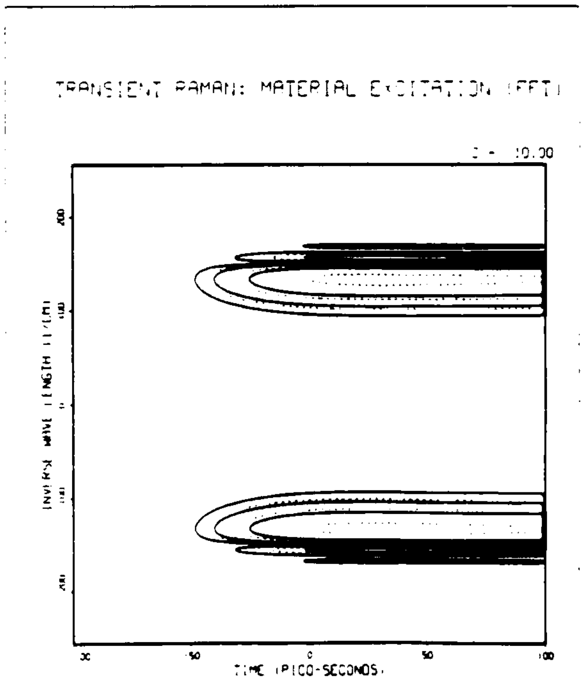


PLT2.DAT (Example C)

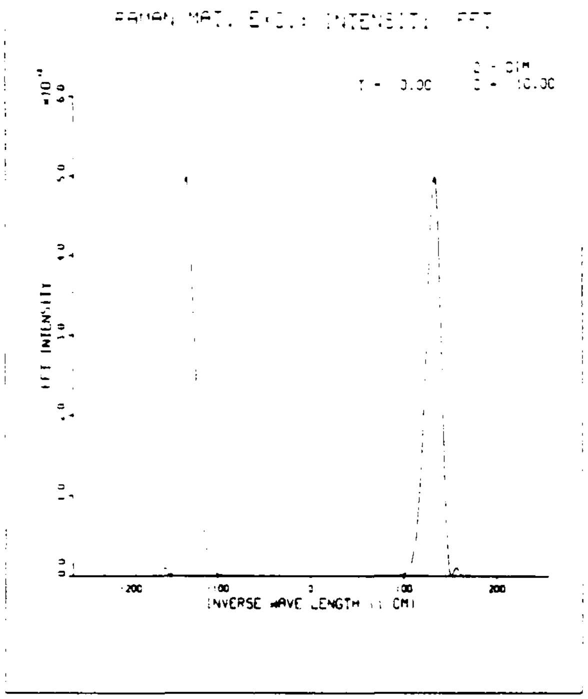


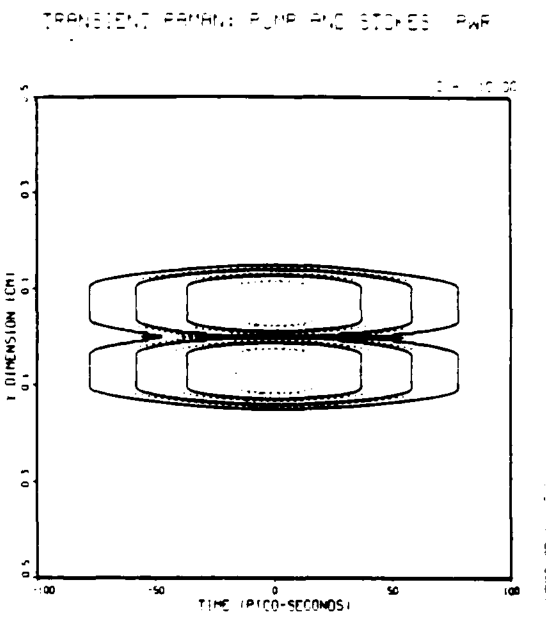


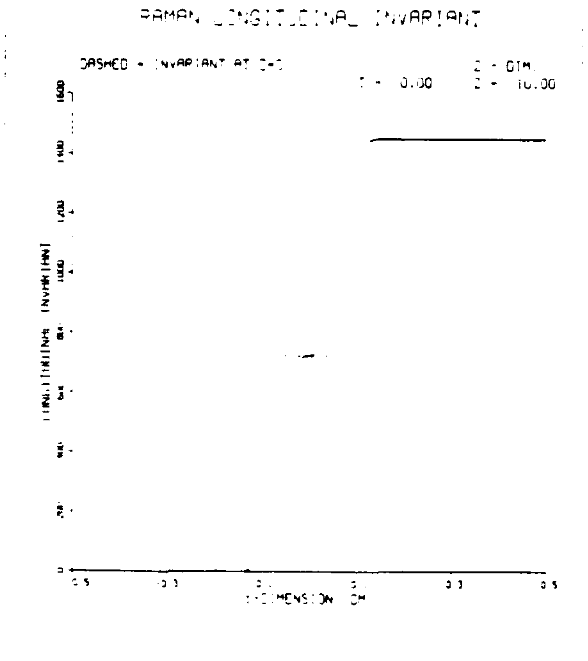


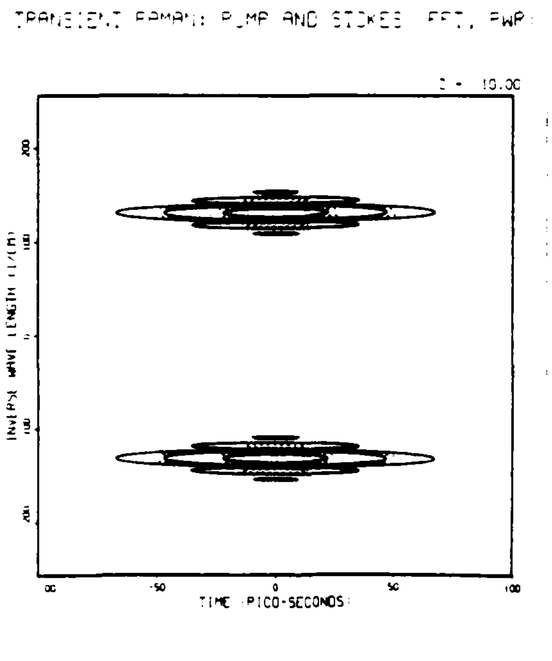


PLT2.DAT (Example C)

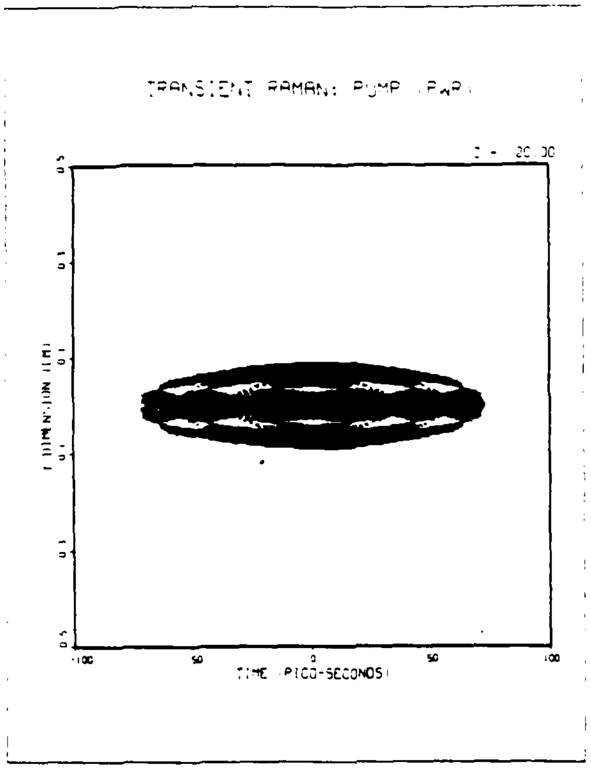


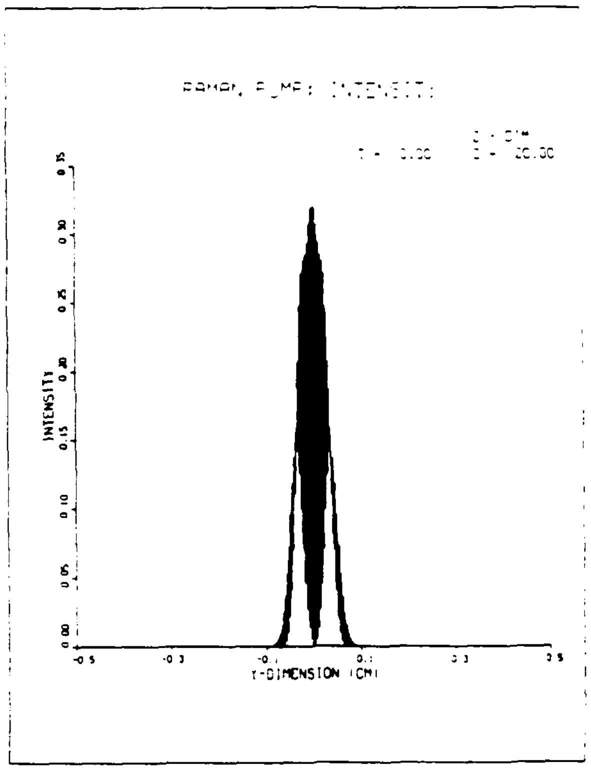


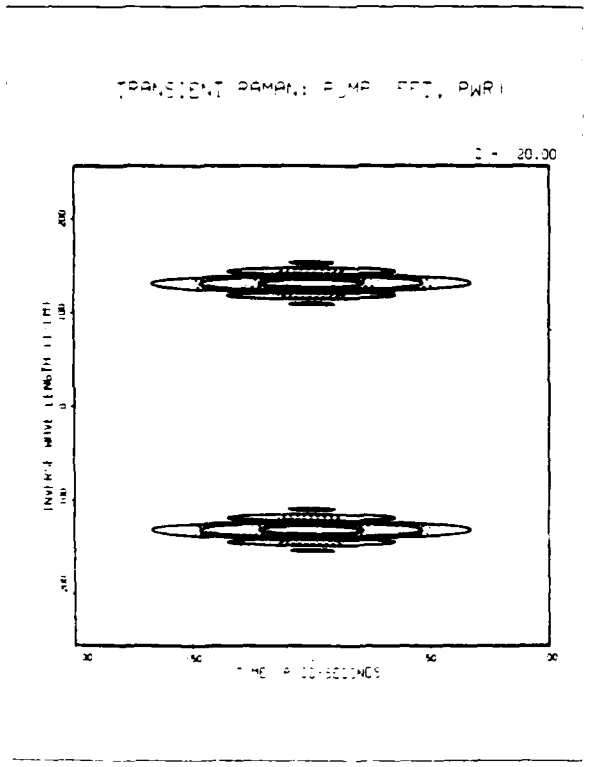


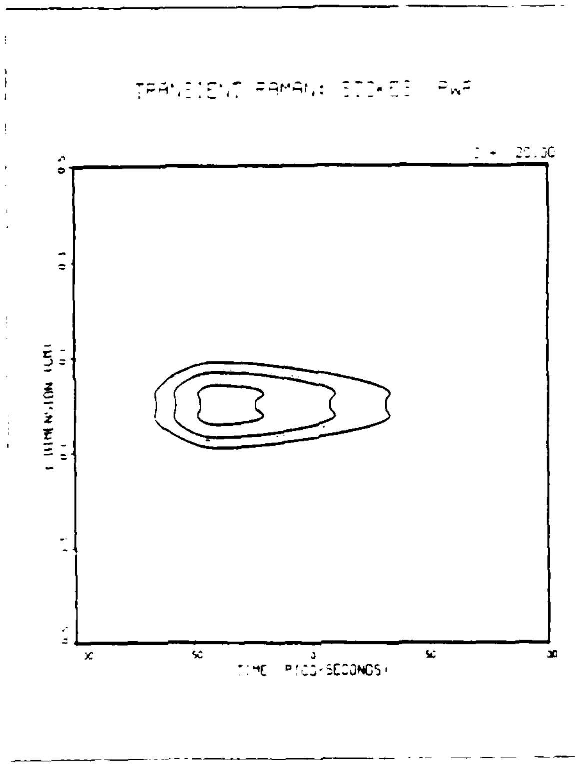


PLT2.DAT (Example C)

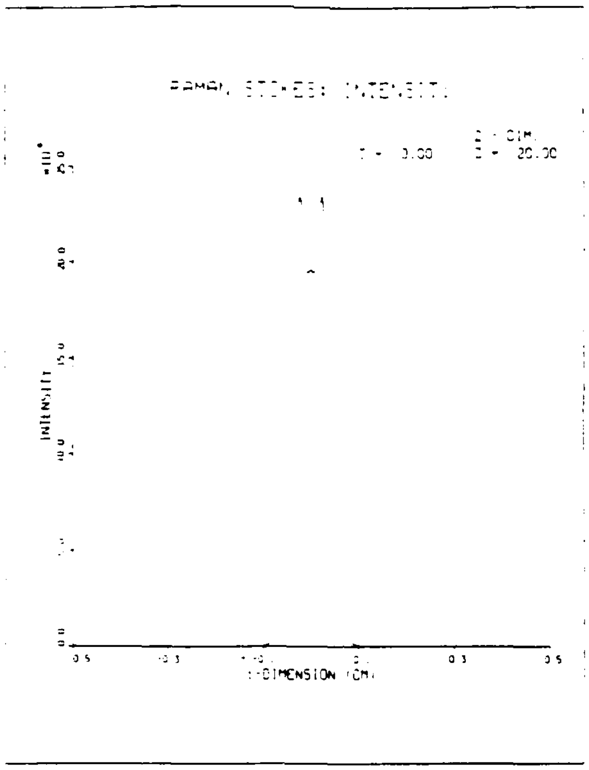


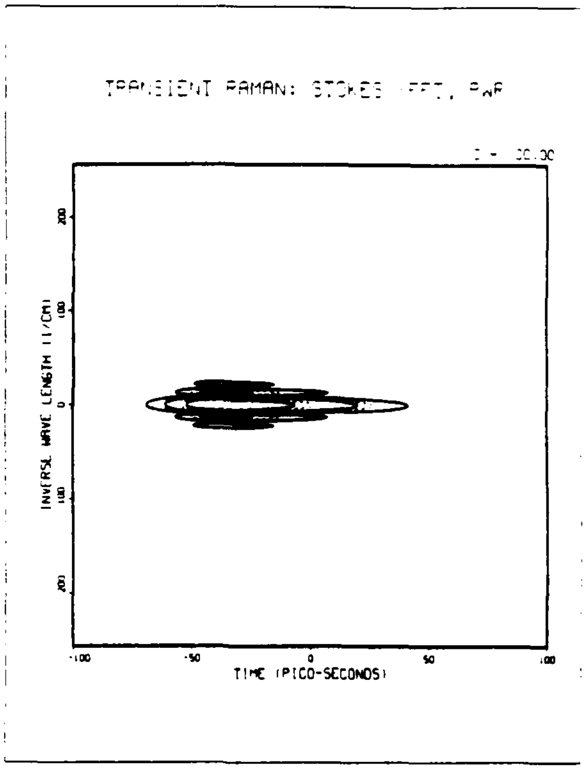


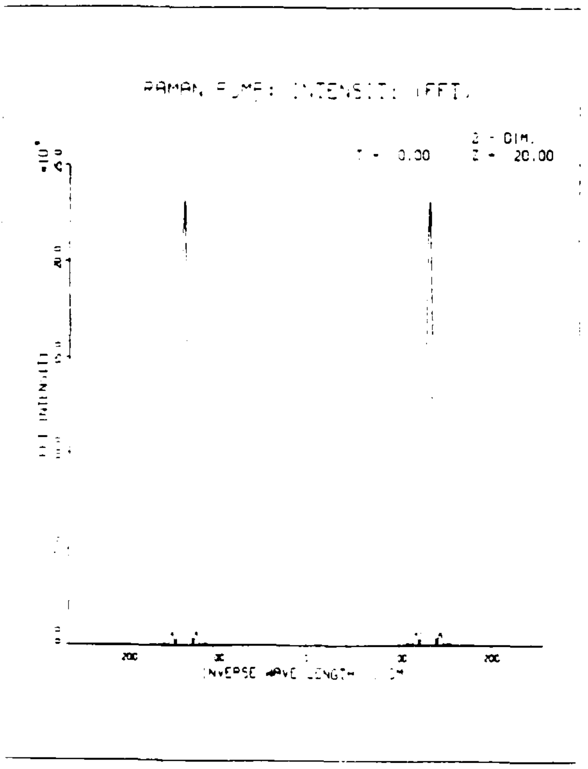


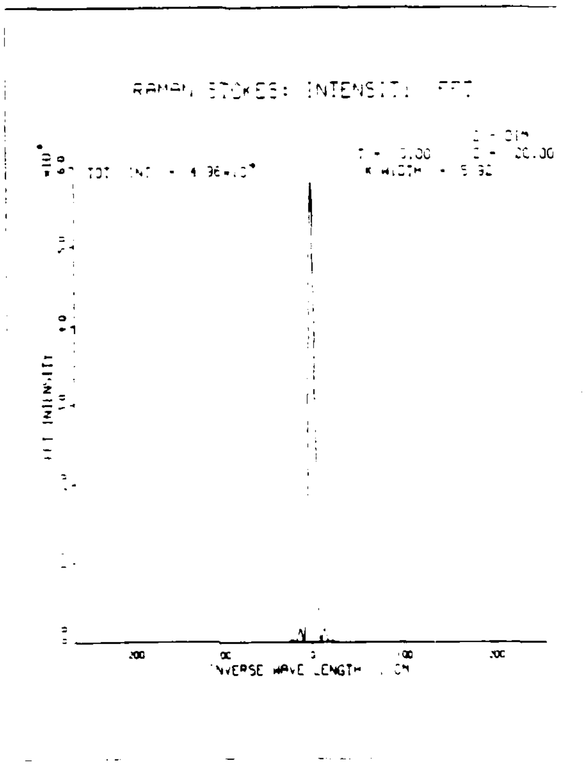


PLT2.DAT (Example C)

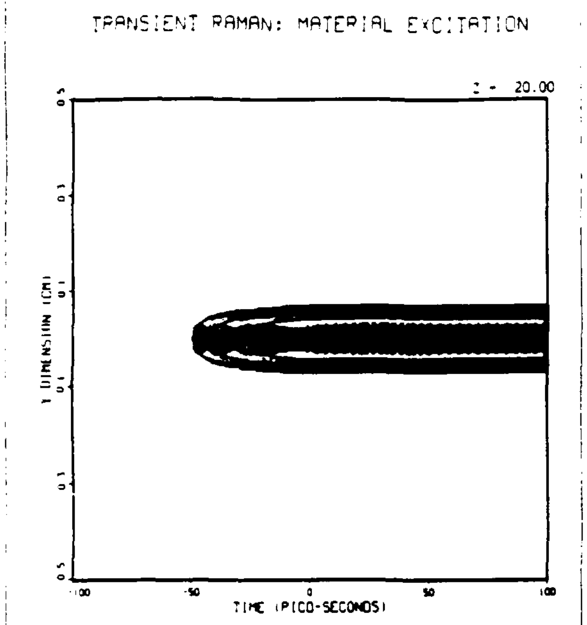


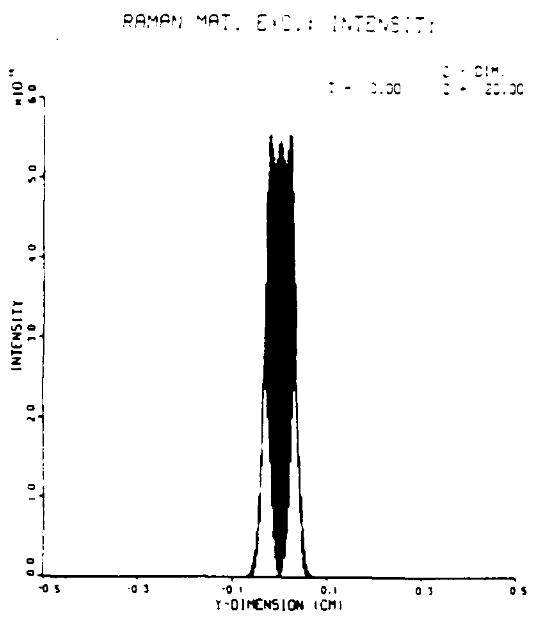


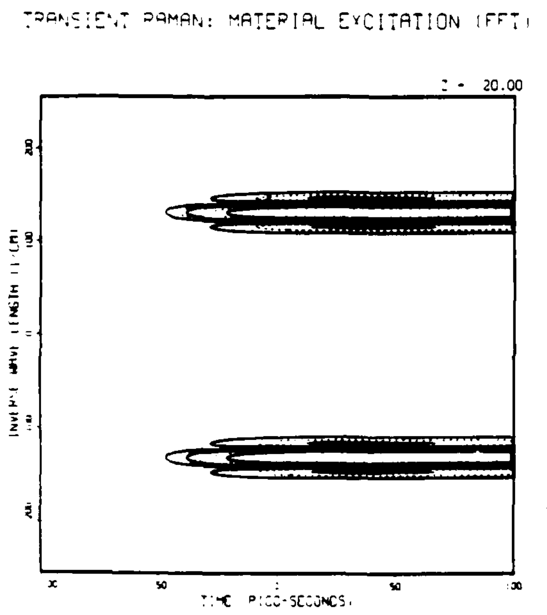


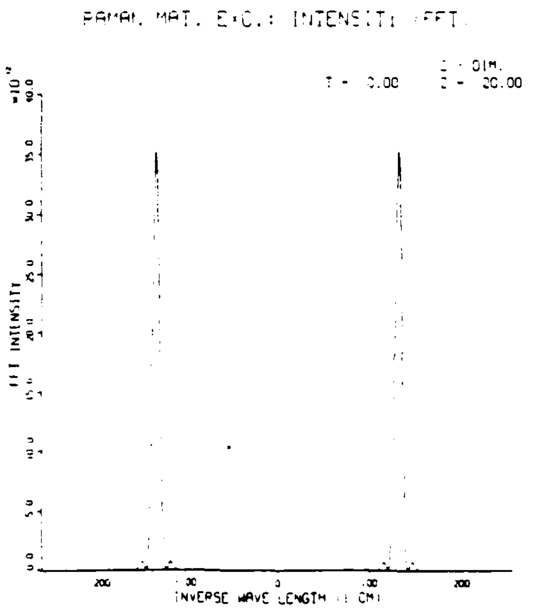


PLT2.DAT (Example C)

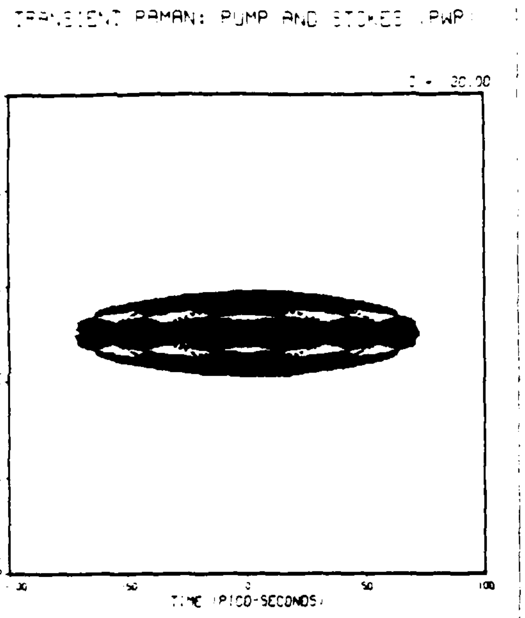


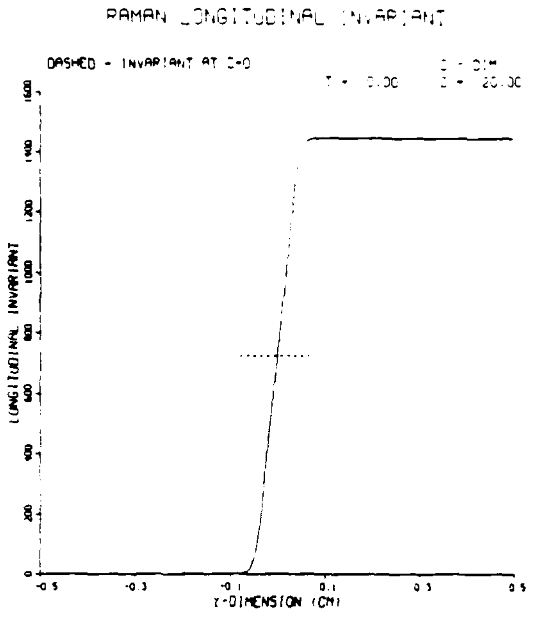


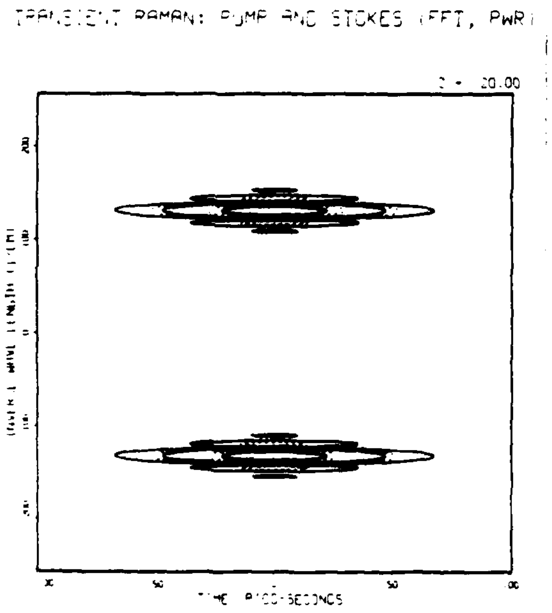


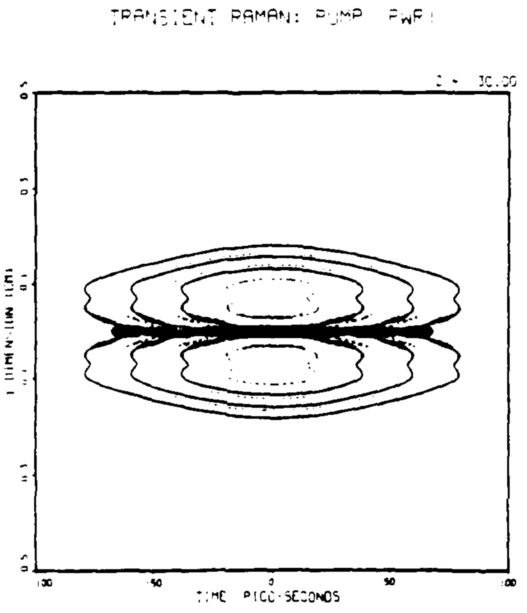


PLT2.DAT (Example C)

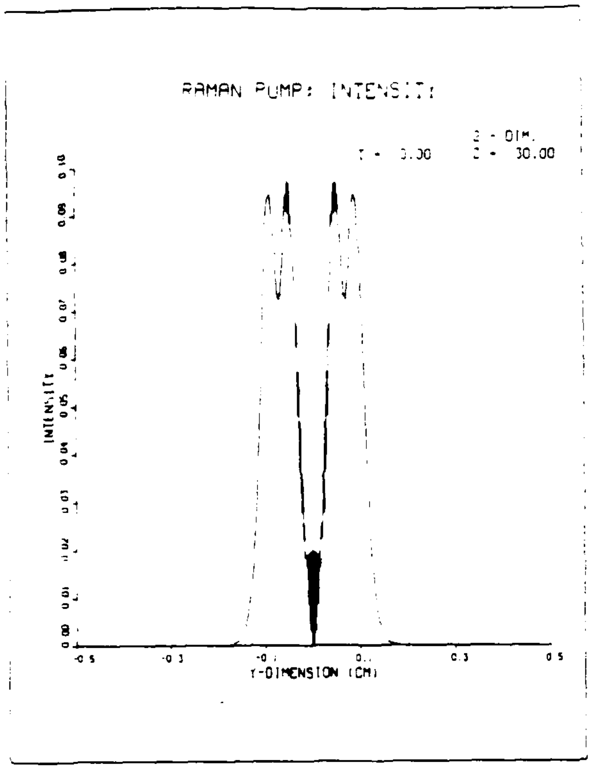


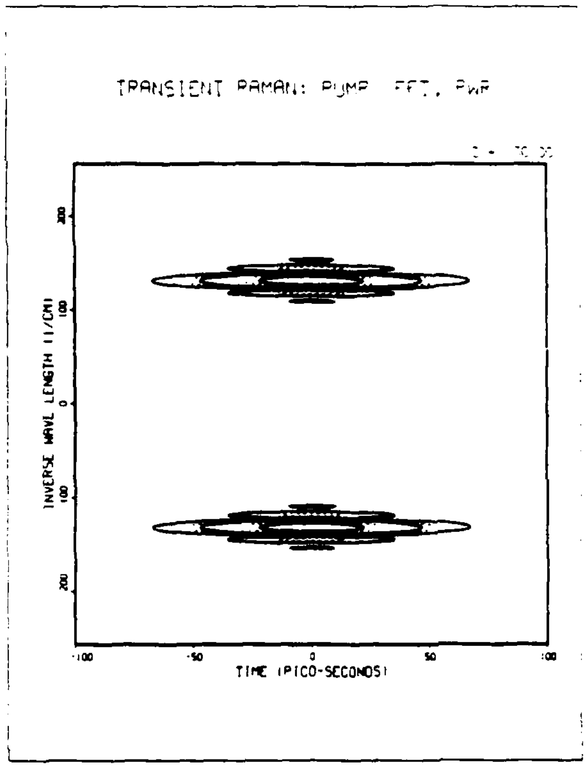


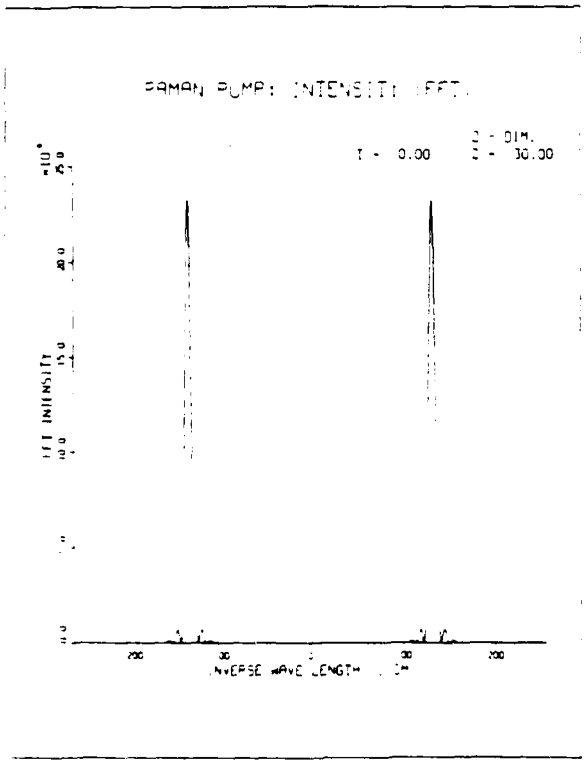


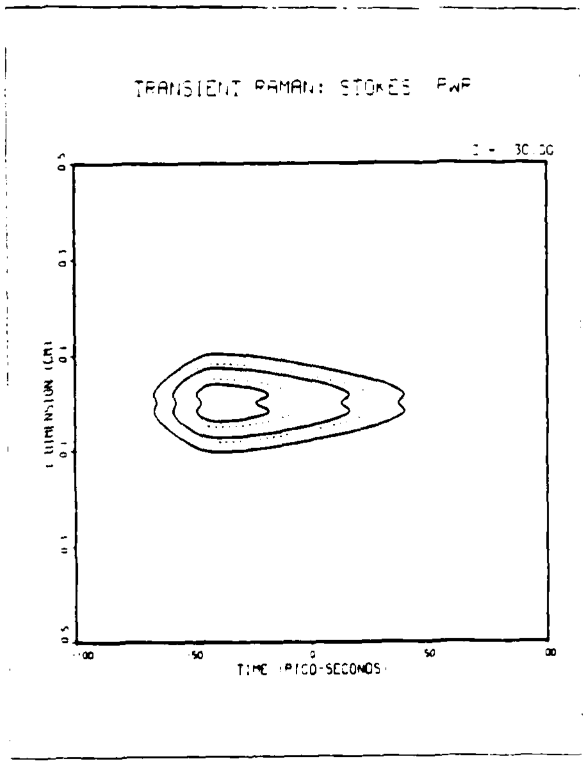


PLT2.DAT (Example C)

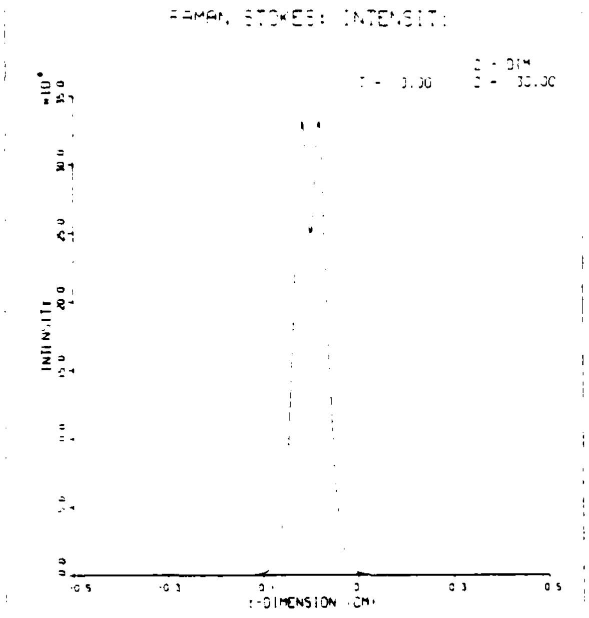


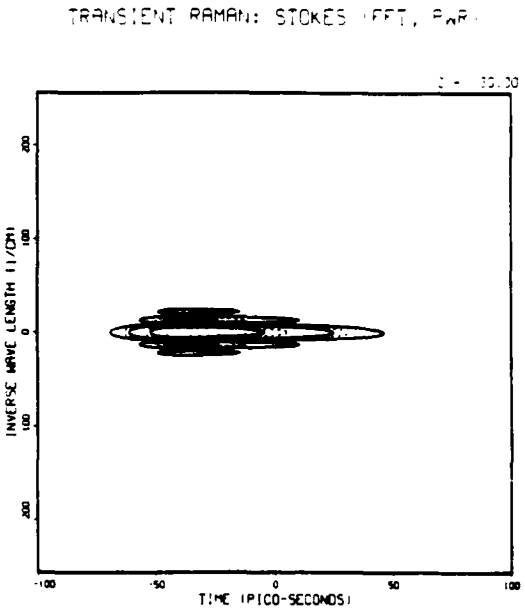


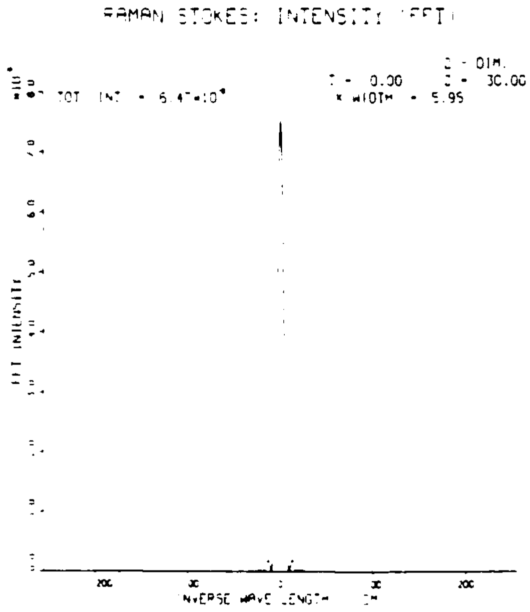


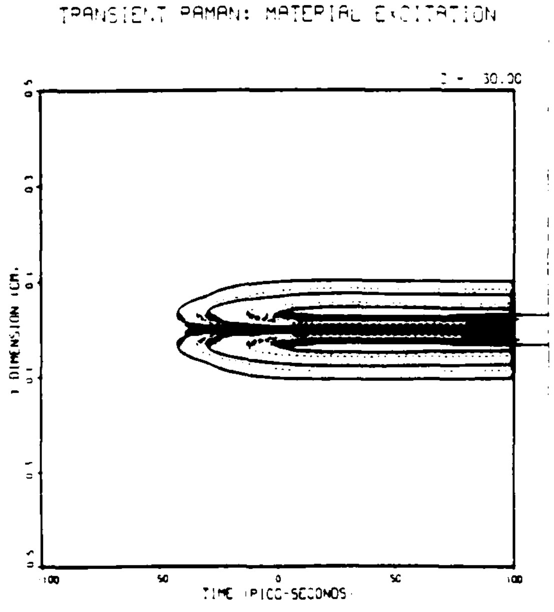


PLT2.DAT (Example C)

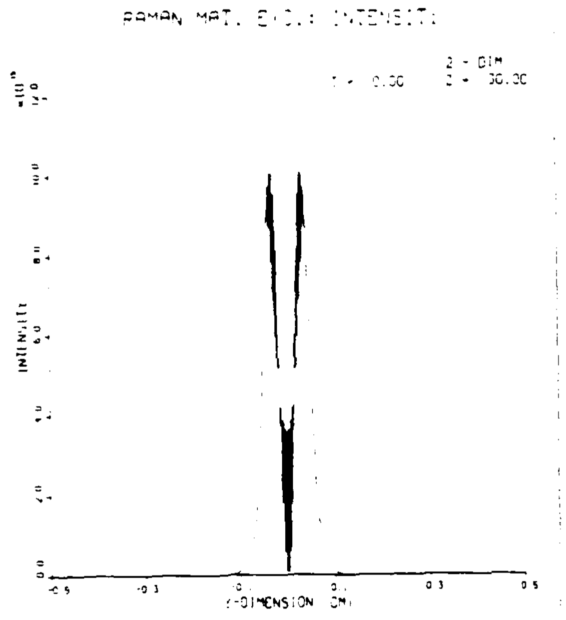


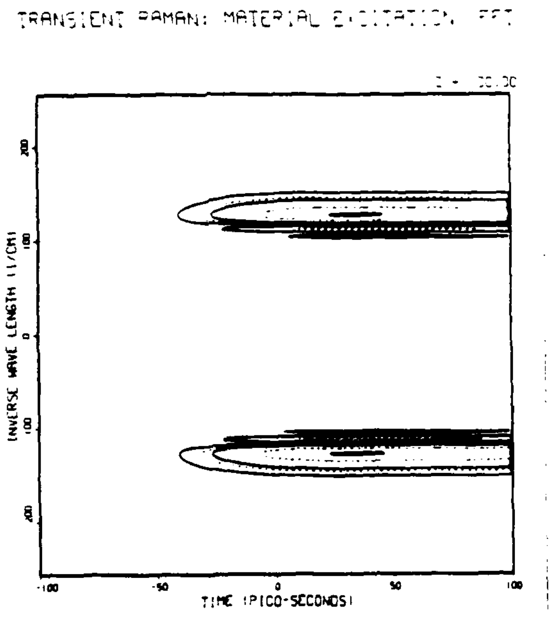


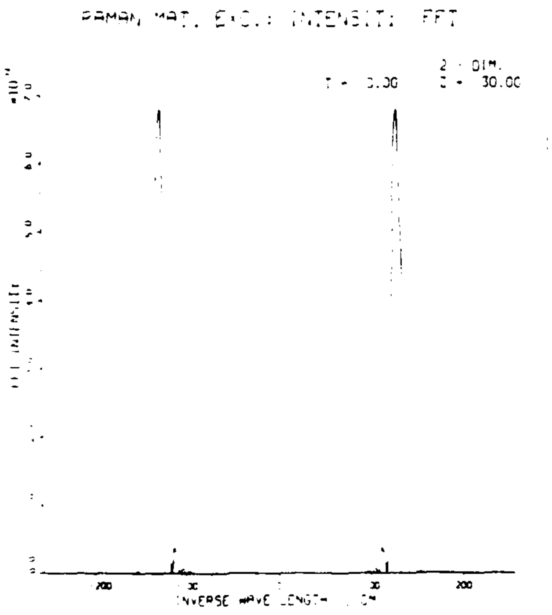


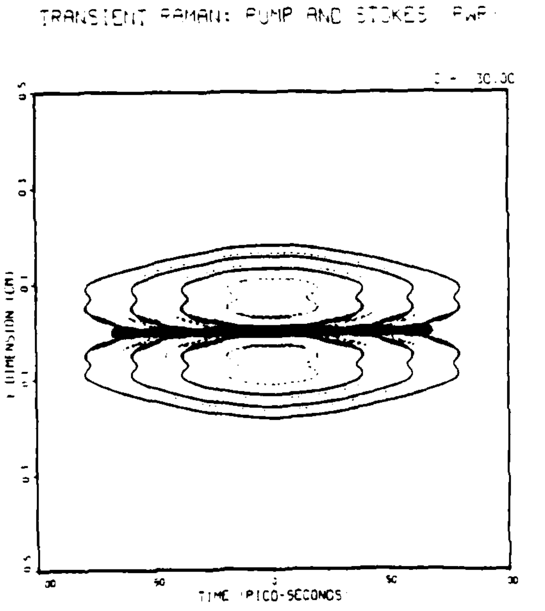


PLT2.DAT (Example C)

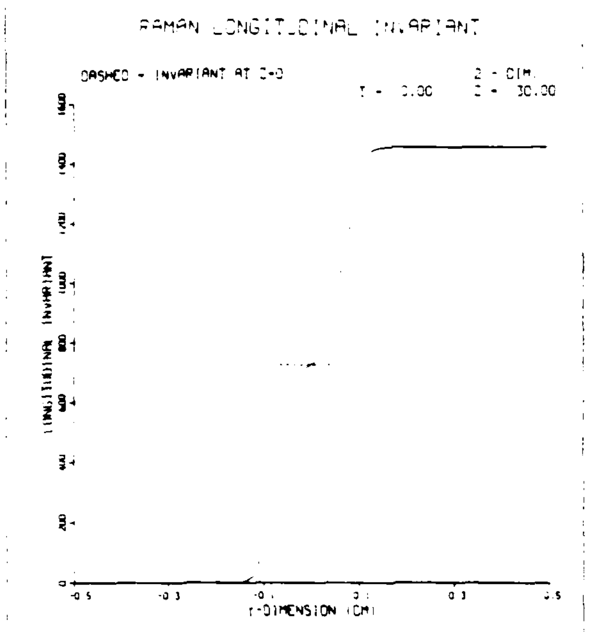


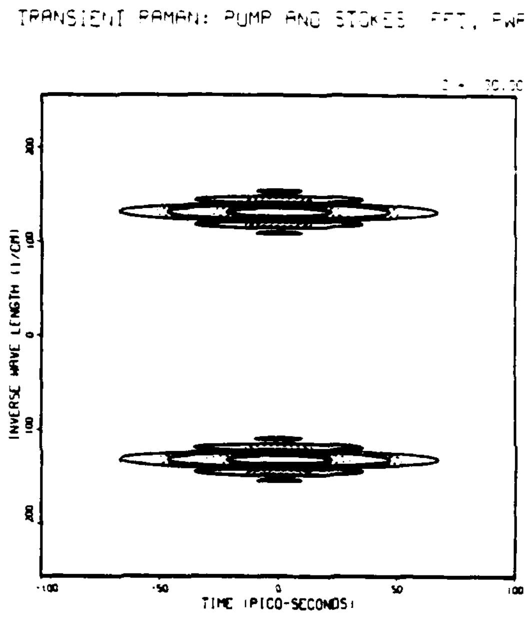


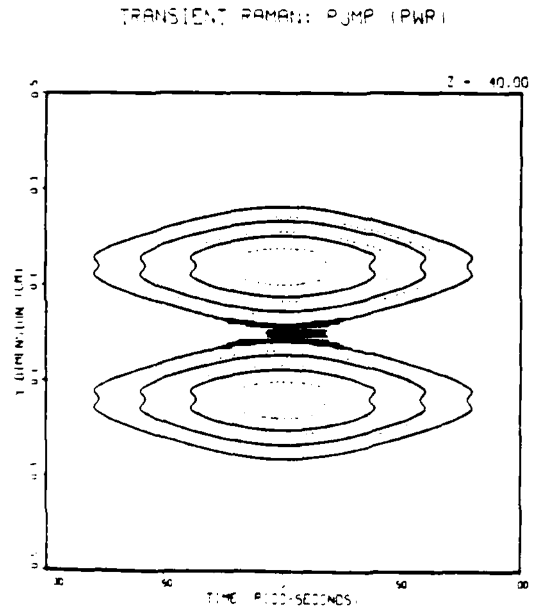


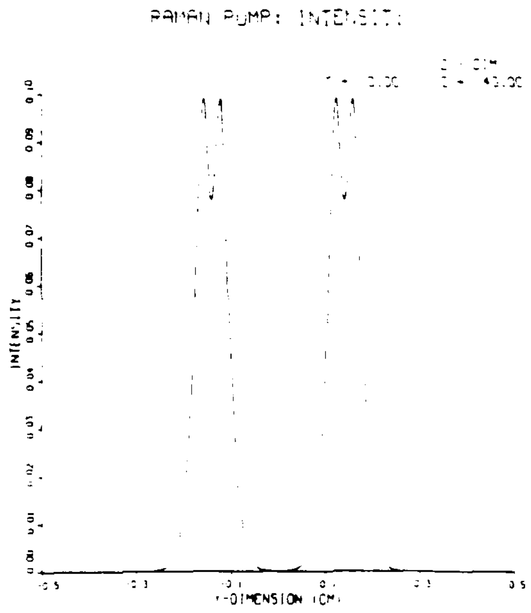


PLT2.DAT (Example C)

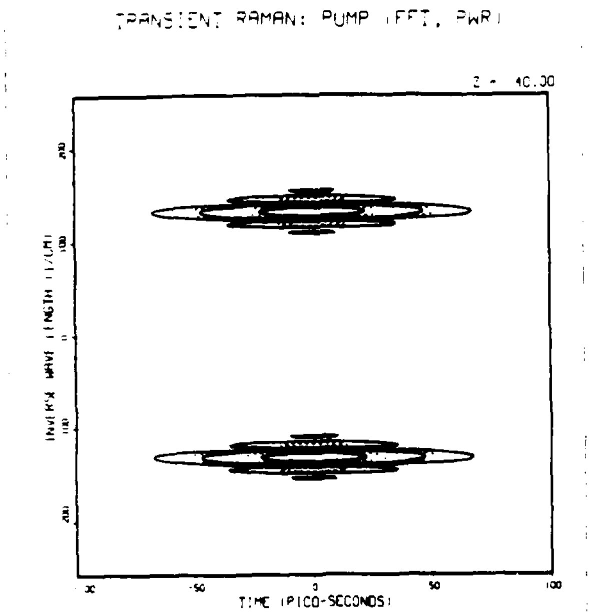


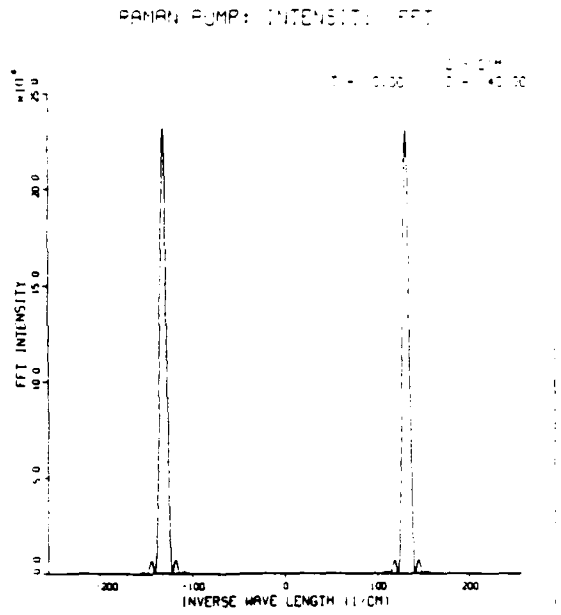


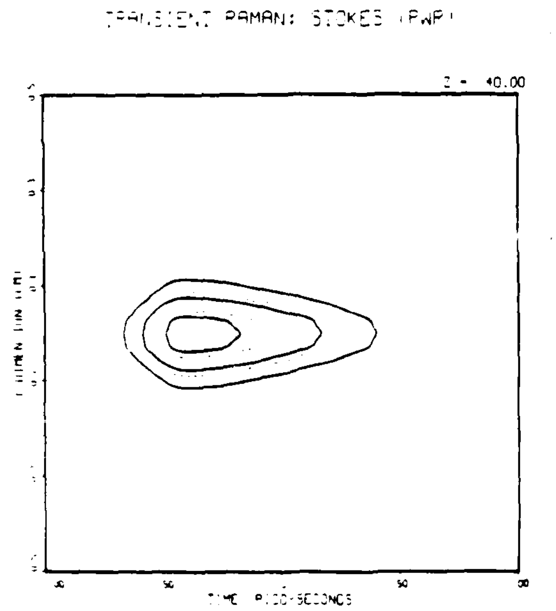


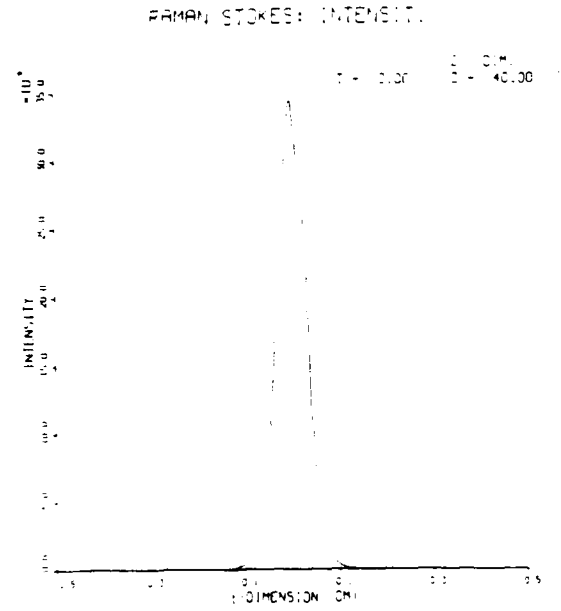


PLT2.DAT (Example C)

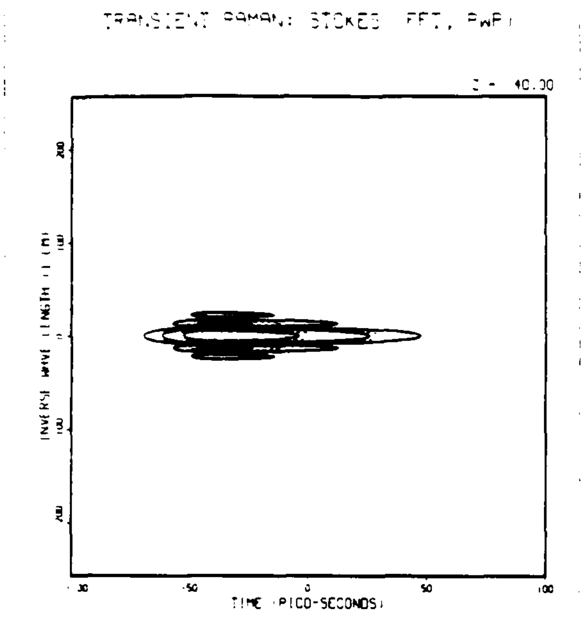


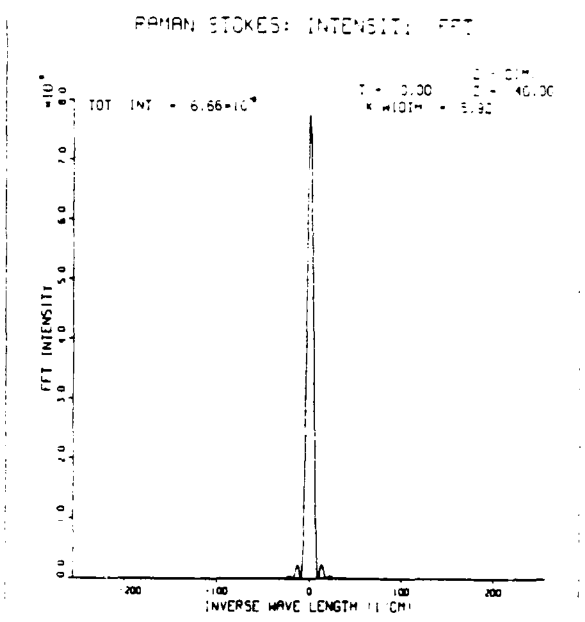


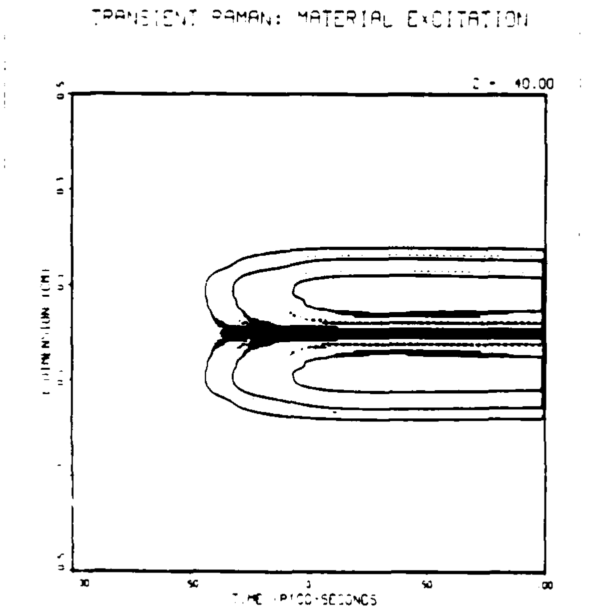


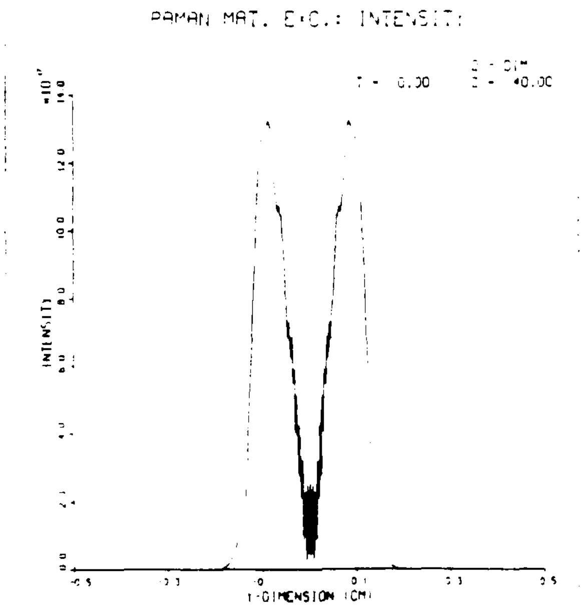


PLT2.DAT (Example C)

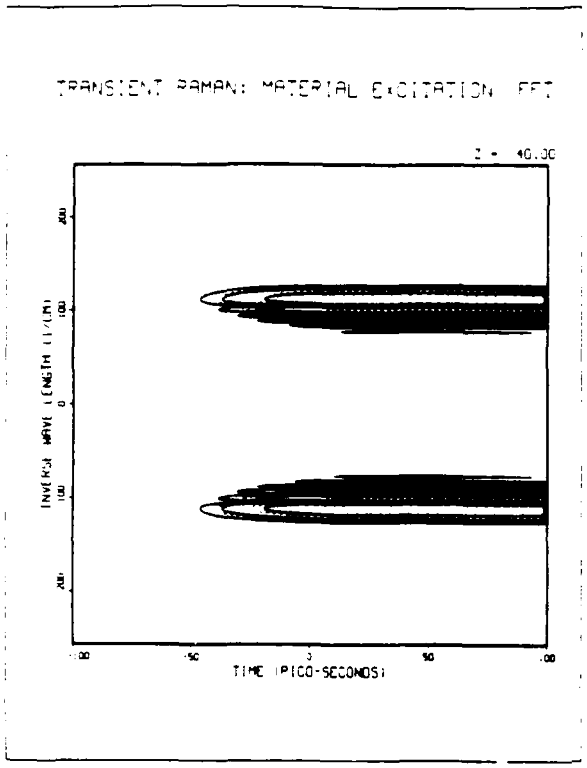


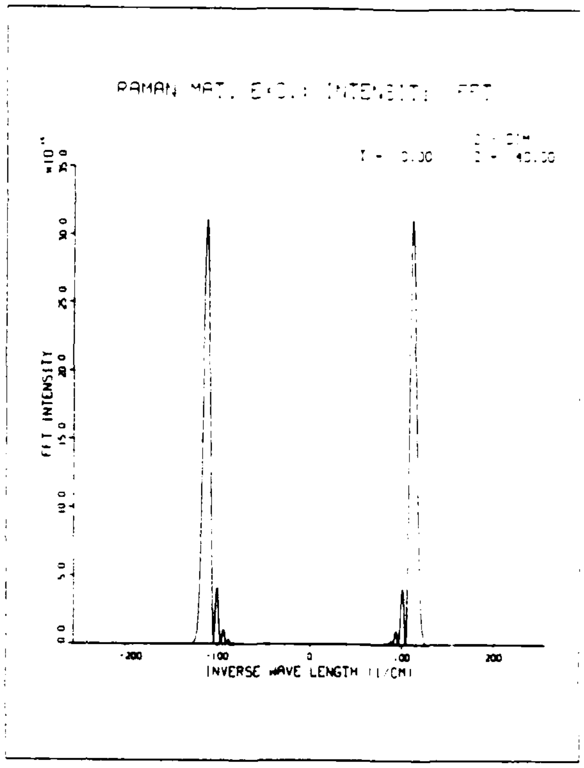


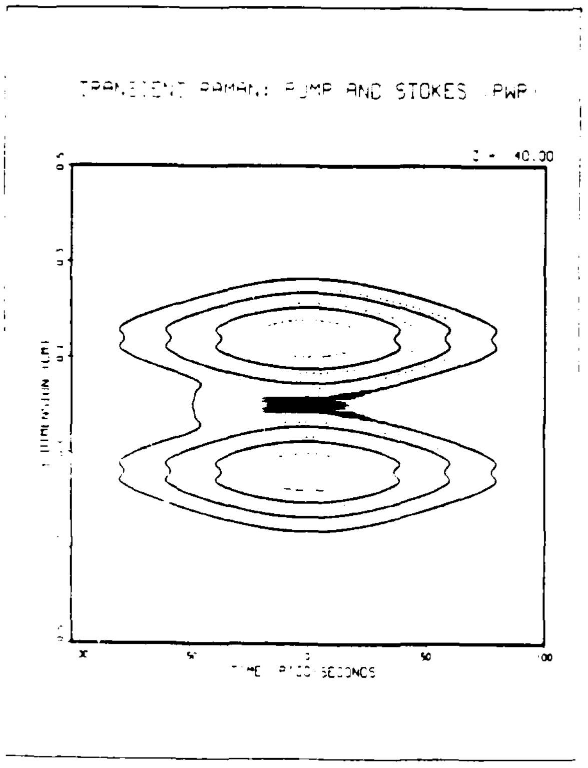


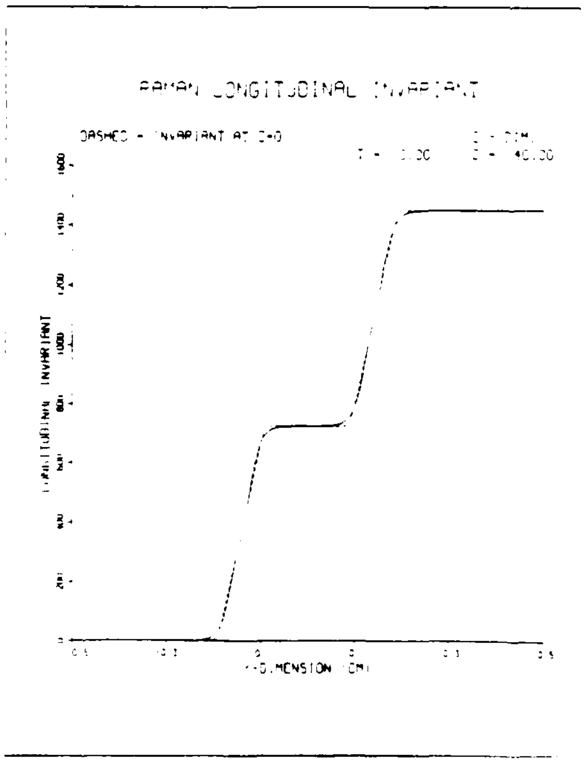


PLT2.DAT (Example C)

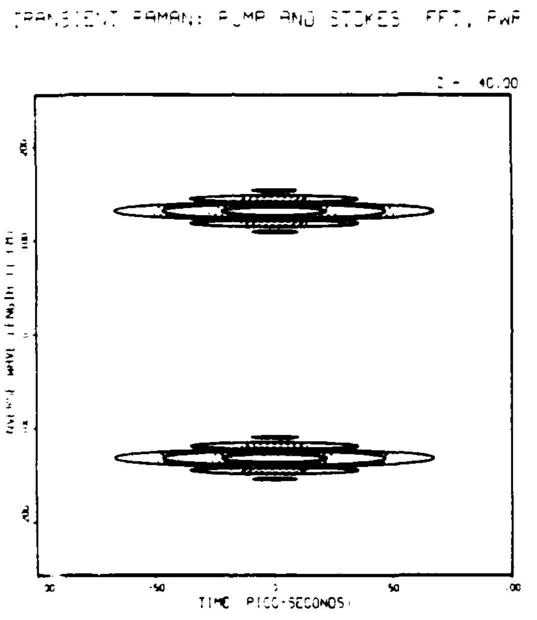








PLT2.DAT (Example C)



APPENDIX D Special Versions of the Codes

It shall be mentioned that several special versions of the presented two programs RAM2D1 and PRAM1 exist. As was mentioned in section IV.B.5, the code RAM2D1C described in this manual is paralleled by the code RAM2D1D. The D version differs from the C version in that it does not keep the field arrays in memory during execution of the program, but ships them in from and out to storage disk as needed.

The versions A and B of RAM2D1 are the same as versions C and D in function, but different in form. They are adapted for use under the CTSS operating system as installed in the National Magnetic Fusion Energy Computing Center (NMFECC) at the Lawrence Livermore National Laboratory (LLNL), CA, where the code was implemented first. The A and B version take the CIVIC FORTRAN compiler while the C and D versions ought to be compiled by the CFT compiler.

Since the data from either RAM2D1A or RAM2D1B have the same format, one version of the diagnostic program PRAM1 is sufficient. This is called PRAM1AB. In connection with the theoretical studies just mentioned, several special adaptations of PRAM1AB span off. These are PRAM1A, PRAM1B, PRAM1C, PRAM1D, and PRAM1E. The output data files from the NRL-based RAM2D1C and RAM2D1D are also identical in form and are diagnosed with PRAM1CD which is described in this manual.

A special version of RAM2D1A is called RMS1DT which is basically identical with RAM2D1A, but contains an additional block of code that is executed when the program runs in the one-dimensional transient limit. In this limit, an extra output file with time history data on the fields is created. This version was used for obtaining comparison with the results of a theory pertinent to strong pump depletion developed by the authors. The associate diagnostic program of RMS1DT is PRAM1E which plots the number of depletion holes in the pump and the ratio of initial to final pump energy. An expanded version of PRSE is PR1ENL which calculates and plots also other aspects of the analytical theory.

APPENDIX C

Publications

"Application of Lie methods to autonomous Hamiltonian perturbations of the Korteweg-de Vries equation: Second-order calculation," (C.R. Menyuk), in *Nonlinear Evolutions*, J.J.P. Léon, ed. (World Scientific Publ., Singapore, 1988), pp. 571-592.

APPLICATION OF LIE METHODS TO AUTONOMOUS HAMILTONIAN PERTURBATIONS OF THE KORTEWEG-DE VRIES EQUATION: SECOND ORDER CALCULATION

C.R. Menyuk
Department of Electrical Engineering
University of Maryland
Catonsville, MD 21228
and
College Park, MD 20742
USA

ABSTRACT

The Lie perturbation method of Hori and Deprit is a practical method for determining the evolution of nearly integrable finite-dimensional Hamiltonian systems. I show how to extend this approach to small, autonomous Hamiltonian perturbations of the Korteweg-de Vries equation. Explicit second order calculations are carried out in some simple cases where the initial data contains a single solitary wave and a small amount of radiation. We show explicitly that a solitary wave will emerge from the initial data through second order. This approach can be extended to arbitrarily high order.

I. INTRODUCTION

Nonlinear wave equations which can be integrated using spectral methods are quite special. Nonetheless, they play an important role in physics and have been used to model a wide variety of phenomena. Generally, these equations are derived by making a small parameter expansion of the underlying physical equations. At zeroth order, one obtains the linear equation. Moving into the wave frame, one obtains the integrable wave equation at first order. If this process is continued to higher order, one obtains corrections which in general destroy the equation's integrability[1-4]. The two most experimentally important examples which lead to the Kortweg-de Vries equation are water waves in channels[1,5,6] and ion acoustic waves in plasmas[2,7,8]. In the first case, the underlying physical equation is Euler's equation with appropriate boundary conditions, and small parameters include d/h, the height of the pulse divided by the height of the channel, and h/l, the height of the channel divided by the length of the pulse. In the second case, the underlying equations are the two fluid equations with inertia-less electrons and constant temperatures. Small parameters include $\delta n/n$, the size of the ion density pulse divided by the undisturbed ion density, and T_i/T_e, the ratio of the ion and electron temperatures.

Experimentally, one finds that these small parameters can be quite large, as large as 0.3–0.4 using standard normalizations, and solitons (or more precisely solitary waves) are seen to emerge from an initial pulse just as if the system was integrable; however, their widths and velocities are related to their heights somewhat differently than in the Korteweg-de Vries equation[5–8]. By contrast, relatively small dissipative perturbations—or perturbations that vary in space and time—are sufficient to destroy the integrable-appearing behavior.

In the past I have tried to provide qualitative insight into this behavior by showing that the higher order corrections yield Hamiltonian perturbations that do not vary in space and time[3,4], *i.e.* autonomous Hamiltonian perturbations, and that under certain conditions, which the experiments reproduce reasonably well, solitary waves emerge to all orders in the small parameters[9–12]. There are important restrictions: First, an asymptotic theory in which secularities are removed order by order can only be carried out once the solitons corresponding to the poles of the spectral data are well-separated. Previous to this separation, the solitary waves interact and continuum radiation and even new solitary waves can be produced. By reducing the perturbation strength, the amplitudes of any new solitary waves produced and their number can be bounded. A second restriction is that the theory in its present form is non-uniform in x, the coordinate space. As a consequence, the possibility cannot be ruled out that a portion of the continuum "to all orders" might actually be a low, broad solitary wave "beyond all orders." The converse also holds. A third restriction is that we consider initial data which falls off faster than some exponential as $x \to \pm\infty$ and which is analytic in some strip surrounding the real axis in complex x-space.

Despite these restrictions, it is my hope that this perturbative approach will prove quantitatively useful in the long run. While it is simpler to determine solitary wave solutions by looking for stationary solutions of the equations, such an approach does not allow one to determine the amplitude(s) of the solitary wave(s) which will ultimately emerge from given initial data. The approach presented here does.

In this work, I will be concentrating on the experimentally important case where the initial data contains only a single solitary wave, or, more precisely, only a single pole in the transmission data. We thus avoid any problems related to solitary wave interactions. I will be using a Hamiltonian approach, specifically the Lie approach first developed by Hori[13] and Deprit[14]. I use a Hamiltonian approach, rather than a more general approach such as that of Karpman and Maslov[15] or Kaup and Newell[16], because it allows one to concentrate in a natural way on the autonomous Hamiltonian systems and obtain results which only apply to them. I use the Lie approach rather than the Poincaré-von Zeipel approach because the Lie approach is now generally considered the simpler of the two to use[17].

I will be concentrating in this paper on perturbed Hamiltonians of the form

$$H[u] = H_0[u] + \epsilon H_1[u], \tag{1}$$

where

$$\begin{aligned} H_0[u] &= \int_{-\infty}^{\infty} dx\,(u^3 + \frac{u_x^2}{2}), \\ H_1[u] &= \int_{-\infty}^{\infty} dx\, u^p, \end{aligned} \tag{2}$$

with $p = 2, 3, 4$ and 5. Using the Poisson bracket

$$[F,G] = \int_{-\infty}^{\infty} dx \left(\frac{\delta F}{\delta u}\frac{\partial}{\partial x}\frac{\delta G}{\delta u}\right), \tag{3}$$

we find

$$u_t = [u, H] = 6uu_x - u_{xxx} + \epsilon p(p-1)u^{p-2}u_x, \tag{4}$$

which is just the Korteweg-de Vries equation with a small perturbation. The case $p = 2$ corresponds to a Gallilean transformation; the case $p = 3$ corresponds to a change in the nonlinear coefficient of the Kortweg-de Vries equation; and the case $p = 4$ corresponds to a Miura transformation. The case $p = 5$ produces a non-integrable system. For these relatively simple examples, I will calculate the perturbed Hamiltonian through second order, as well as the perturbed potential u through first order.

In previous work, I have studied other simple examples, but only through first order[10]. My motivation for carrying out explicit second order calculations is that some of my colleagues, notably Yuji Kodama, felt that this calculation would be very useful in clarifying the basic structure of the theory by showing in detail how to avoid secularities, as must be done in any Hamiltonian theory. One must divide the Hamiltonian between its coordinate-independent and coordinate-dependent pieces and group the former with the zero-order Hamiltonian at each order. The first non-trivial order at which this separation must be carried out is second order.

In previous work I have primarily employed Hamiltonian perturbation theory to show, with the limitations described earlier, that solitons emerge from arbitrary initial data to all orders in the small parameter for a large class of Hamiltonian perturbations[11,12]. The goal was to obtain insight into the experimentally observed robustness of solitons[5,8]. In this respect my work was motivated by Martin Kruskal's classic study of the theory of adiabatic invariants[18], and, in my opinion, the approach and conclusions are conceptually similar. Nonetheless, it is my hope that this approach will ultimately prove useful in carrying out detailed quantitative comparisons between theory and experiment, much as it has proved useful in the study of satellite orbits about the Earth[13,14], and particle motion in accelerators[17]. In order for the theory to be useful in this regard, one must be comparing to experiments where the perturbations are quite small, the distances quite large, and the measurements quite precise. While experiments modelled by field equations do not seem to fulfill these conditions at present. It is my belief that they will do so within the next twenty years.

The remainder of this paper is organized as follows: In Section II, we specify the action-angle transformation from $u(x)$ to $[p(k), q(k), p_\alpha, q_\alpha]$, the canonical variables which evolve linearly in time. In Section III, we show how to write the Hamiltonian in terms of the canonical variables. In Section IV, we show how to obtain the lowest order Lie generator and discuss the problem of small denominators. Section V contains the explicit calculation of the second order perturbed Hamiltonian and includes the determination of the second order Lie generator. In Section VI, we calculate the first order potential. Section VII contains the conclusions and acknowledgments.

II. ACTION-ANGLE TRANSFORMATION

Before I can apply Hamiltonian methods to the perturbed system, I must determine action-angle variables for the underlying integrable system. Quite generally, we may write the original coordinates as u, where $u = u_i$ in a system with a discrete number of independent variables and $u = u(x)$ in a system with an uncountably infinite number of degrees-of-freedom. We are interested here in the latter case. We are searching for an invertible transformation of the form

$$u \rightarrow (J, \theta), \tag{5}$$

where J and θ represent the ensemble of action and angle variables; in general, J and θ can have both uncountably infinite and discrete components. The set of variables J and θ should be canonically related to each other, and the Hamiltonian should only depend on the action variables J.

The equations of motion in terms of these new variables is trivially integrable. Writing Ω for the ensemble of frequency variables, *i.e.*

$$\Omega_i = \partial H[J]/\partial J_i \qquad \text{and} \qquad \Omega(k) = \partial H[J]/\partial J(k), \tag{6}$$

for the discrete and continuous components, it follows that

$$\begin{aligned} J &= J_0, \\ \theta &= \theta_0 + \Omega t, \end{aligned} \tag{7}$$

where J_0 and θ_0 represent the ensemble of initial conditions. At any time, we may determine $u(x)$ by inverting the action-angle transformation. This situation is shown schematically in Fig. 1. If we take the direct, left-hand path shown as a dashed arrow and integrate the equations of motion using a computer, it is generally necessary to take many small time steps. By contrast, if one integrates the equations of motion using the solid, three-sided path, one can carry out the time integration in one fell swoop. Hence, no matter how complicated the backward and forward transformations, one always "wins" using the three-sided approach over a sufficiently long time interval. I note that this notion of "winning," while useful, is not precise, something which can also be said of the notion of integrable systems. For linear, infinite-dimensional systems, the three-sided path represents Fourier integration; for nonlinear, infinite-dimensional systems, it represents an analogous nonlinear transformation.

The appropriate action-angle transformation when the underlying integrable system is the Korteweg-de Vries equation, was first found by Zakharov and Fadeev[19]. They begin by making the spectral transformation[19]

$$u(x) \rightarrow [r(k), \kappa_j, c_j], \tag{8}$$

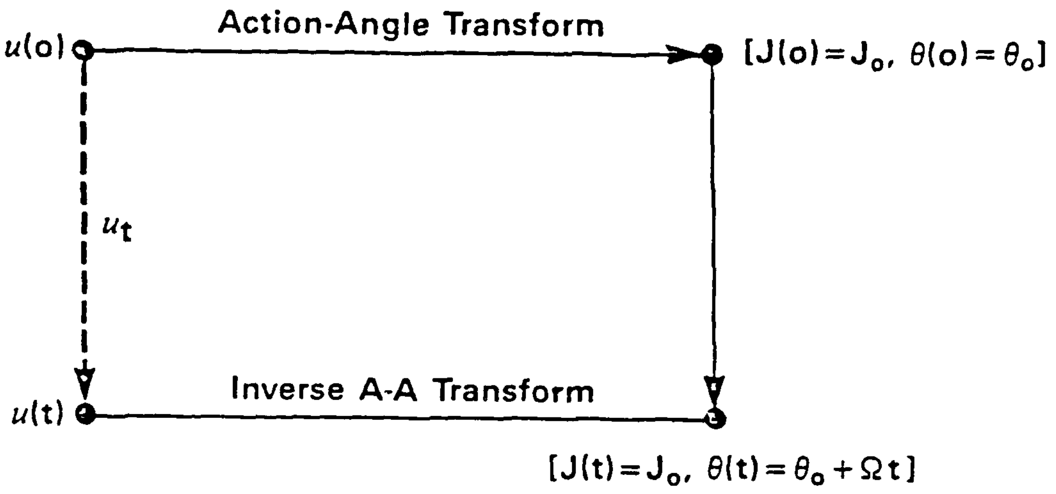

Action-Angle Transform
$u(o)$
$[J(o)=J_o,\ \theta(o)=\theta_o]$
u_t
Inverse A-A Transform
$u(t)$
$[J(t)=J_o,\ \theta(t)=\theta_o+\Omega t]$

FIGURE 1. Schematic illustration of the way in which an action-angle transform and its inverse can be used to solve the equations of motion of a completely integrable system.

and then converting these variables into the appropriate action-angle form. I will make a slightly different choice of variables from theirs which proves to be useful in what follows. Concentrating on the case where the initial data contains a single soliton, my choice is

$$\begin{aligned} p(k) &= -\frac{2k}{\pi}\ln[1-r(k)r(-k)],\\ q(k) &= -\frac{1}{2i}\ln[r(k)/r(-k)],\\ p_\alpha &= \frac{4}{3}\kappa_\alpha^3,\\ q_\alpha &= \frac{1}{\kappa_\alpha}\ln c_\alpha, \end{aligned} \tag{9}$$

where the subscript α indicates the single soliton. The Hamiltonian becomes

$$H_0 = \int_0^\infty dk\, 8k^3 p(k) - \frac{12\cdot 3^{2/3}}{5\cdot 2^{1/3}} p_\alpha^{5/3}, \tag{10}$$

which depends only the action variables, and the Poisson bracket becomes

$$\begin{aligned} [F,G] = \int_0^\infty dk \left[\frac{\partial F}{\partial q(k)}\frac{\partial G}{\partial p(k)} - \frac{\partial F}{\partial p(k)}\frac{\partial G}{\partial q(k)}\right] \\ + \frac{\partial F}{\partial q_\alpha}\frac{\partial G}{\partial p_\alpha} - \frac{\partial F}{\partial p_\alpha}\frac{\partial G}{\partial q_\alpha}, \end{aligned} \tag{11}$$

which is canonical in form. I take the k integrals over the half interval $[0,\infty)$, rather than the full interval $(-\infty,\infty)$, which helps in keeping track of the cross terms which appear in the theory between $-k$ and k. These integrals must eventually be extended first over the full interval and then into the upper half plane. The soliton variables q_α and p_α are related to those used by Zakharov and Fadeev[19] by a simple canonical transformation.

I close this section with an important aside. The canonical form of the Poisson bracket in Eq. (11) follows from the form of the Poisson bracket in Eq. (3). Once the Korteweg-de Vries equation is perturbed, it is not evident *a priori* that this symplectic structure will suffice at all orders. Recently, H. H. Chen and I[3,4] have shown that it does indeed suffice in cases where the physically important systems of one-dimensional ion acoustic waves or shallow channel water waves are being considered.

III. DETERMINING THE HAMILTONIAN

My next task is to re-express the Hamiltonian, $H[u] = H_0[u] + \epsilon H_1[u]$ explicitly in terms of the canonical variables. Zakharov and Fadeev have showed us how to obtain an explicit expression for $H_0[u]$, and the result is given by Eq. (10). Their procedure, however, cannot be applied to the general case where the Hamiltonian $H_1[u]$ will depend on the canonical coordinates as well as the momenta. Instead, I directly calculate $H_1[u]$. I begin by determining u in terms of the canonical variables. In the case of interest here where u contains a single solitary wave (or more precisely the transmission coefficient has a single pole at the initial time), I write

$$u = u_\alpha + 2\frac{d}{dx}K(x,x), \tag{12}$$

where

$$u_\alpha = -2\kappa_\alpha^2 \operatorname{sech}^2[\kappa_\alpha(x + q_\alpha/2)]$$

is the single soliton solution and $K(x,y)$ is given by the solution to the Marchenko equation

$$K(x,y) + F(x,y) + \int_{-\infty}^{x} dz\, K(x,z)F(z,y) = 0 \quad . \tag{13}$$

The kernel $F(x,y)$ is given by the relation

$$F(x,y) = \int_{-\infty}^{\infty} \frac{dk}{2\pi}\, r(k) g_\alpha(x,k) g_\alpha(y,k), \tag{14}$$

where

$$g_\alpha(x,k) = \exp(-ikx)\left(\frac{k - i\kappa_\alpha \tanh[\kappa_\alpha(x + q_\alpha/2)]}{k + i\kappa_\alpha}\right) \tag{15}$$

is the left Jost function corresponding to a single soliton potential. The Neumann expansion of Eq. (13)

$$K(x,y) = -F(x,y) + \int_{-\infty}^{x} dz\, F(x,z)F(z,y) - \cdots, \tag{16}$$

is always convergent. This Neumann expansion is essentially an expression in powers of $r(k)$. Writing now,

$$u = u_\alpha + \mathrm{u}_1 + \mathrm{u}_2, \tag{17}$$

through second order in powers of $r(k)$, and letting

$$\xi = x + q_\alpha/2, \tag{18}$$

I find explicitly

$$\mathrm{u}_1 = 4\int_{-\infty}^{\infty} \frac{dk}{2\pi}\, r(k)\exp(ikq_\alpha)\frac{\exp(-2ik\xi)}{k + i\kappa_\alpha} \sum_{j=0}^{1}\Big[a_j \operatorname{sech}^{2j}(\kappa_\alpha\xi) + b_j \operatorname{sech}^{2j}(\kappa_\alpha\xi)\tanh(\kappa_\alpha\xi)\Big], \tag{19a}$$

$$\mathrm{u}_2 = -2\int_{-\infty}^{\infty} \frac{dk_1}{2\pi}\, r(k_1)\int_{-\infty}^{\infty} \frac{dk_2}{2\pi}\, r(k_2)\frac{\exp[i(k_1+k_2)q_\alpha]\exp[-2i(k_1+k_2)\xi]}{(-i)(k_1 + i\kappa_\alpha)^2(k_2 + i\kappa_\alpha)^2} \sum_{j=0}^{2}\Big[c_j \operatorname{sech}^{2j}(\kappa_\alpha\xi) + d_j \operatorname{sech}^{2j}(\kappa_\alpha\xi)\tanh(\kappa_\alpha\xi)\Big], \tag{19b}$$

where

$$\begin{aligned}
a_0 &= ik(k^2 - \kappa_\alpha^2),\\
b_0 &= 2k^2\kappa_\alpha,\\
a_1 &= 2ik\kappa_\alpha^2,\\
b_1 &= \kappa_\alpha^3,\\
c_0 &= 2i(k_1+k_2)[(k_1k_2 - \kappa_\alpha^2)^2 - (k_1+k_2)^2\kappa_\alpha^2],\\
d_0 &= 4(k_1+k_2)^2(k_1k_2 - \kappa_\alpha^2)\kappa_\alpha,\\
c_1 &= 2i(k_1+k_2)[(k_1+k_2)^2 + (2k_1k_2 - 3\kappa_\alpha^2)]\kappa_\alpha^2,\\
d_1 &= 2[2(k_1+k_2)^2 + (k_1k_2 - \kappa_\alpha^2)]\kappa_\alpha^3,\\
c_2 &= 3i(k_1+k_2)\kappa_\alpha^4,\\
d_2 &= 0.
\end{aligned} \tag{20}$$

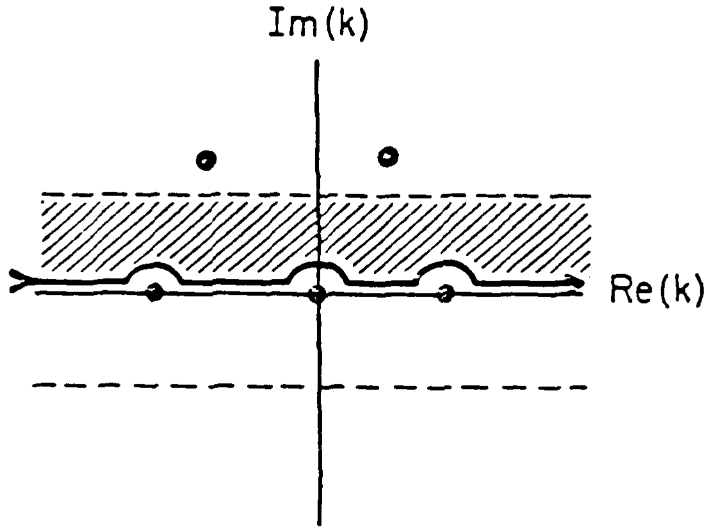


FIGURE 2. Schematic illustration of an integration path through the upper half Bargmann strip, represented by the hatched region. The dots represent poles.

In calculating Eq. (19b), I had to exchange an integral of the form $\int_{-\infty}^{z} dz \cdots$ with two integrals of the form $\int_{-\infty}^{\infty} dk \cdots$. The integral over z was then explicitly evaluated. This exchange is only valid when $Im(k) > 0$, *i.e.*, when the integrals over k are performed in the upper half Bargmann strip shown schematically as the hatched area in Fig. 2. While quite simple conceptually, this result is important as it tells me how to integrate around the poles which appear in the theory on the real k-axis.

At this point, I determine

$$H_1[u] = \int_{-\infty}^{\infty} u^p d\xi \tag{21}$$

in terms of the canonical variables. To do so, I need to exchange the integral $\int_{-\infty}^{\infty} d\xi \cdots$ with the integrals over k. This exchange is not permitted unless the integrand decreases exponentially as $\xi \to +\infty$ which is not always the case. When that is not the case, we may pick up a δ-function contribution. To show how this works, I first consider the case $p = 2$. In this case

$$u^2 = u_\alpha^2 + 2u_\alpha u_1 + u_1^2 + 2u_2 u_\alpha; \tag{22}$$

so, writing

$$H_1 = h_0 + h_1 + h_2, \tag{23}$$

where I have expanded H_1 in powers of $r(k)$ through second order, I find

$$\mathrm{h}_0 = \int_{-\infty}^{\infty} u_\alpha^2 \, d\xi \; = \; 4\kappa_\alpha^4 \int_{-\infty}^{\infty} \operatorname{sech}^4(\kappa_\alpha \xi)\, d\xi \; = \; \frac{16}{3}\kappa_\alpha^3. \tag{24}$$

I also find

$$\mathrm{h}_1 = \int_{-\infty}^{\infty} 2u_\alpha \mathrm{u}_1 \, d\xi \tag{25}$$

which is non-singular and yields

$$\begin{aligned} \mathrm{h}_1 \; = \; -16 \int_{-\infty}^{\infty} \frac{dk}{2\pi}\, r(k) \exp(ikq_\alpha) \int_{-\infty}^{\infty} d\xi \, \frac{\exp(-2ik\xi)}{(k+i\kappa_\alpha)^2} \\ [ik(k^2-\kappa_\alpha^2)\kappa_\alpha^2 \operatorname{sech}^2(\kappa_\alpha\xi) + 2k^2\kappa_\alpha^3 \operatorname{sech}^2(\kappa_\alpha\xi)\tanh(\kappa_\alpha\xi) \\ + 2ik\kappa_\alpha^4 \operatorname{sech}^4(\kappa_\alpha\xi) + \kappa_\alpha^5 \operatorname{sech}^4(\kappa_\alpha\xi)\tanh(\kappa_\alpha\xi)] = 0. \end{aligned} \tag{26}$$

At next order,

$$\mathrm{h}_2 = \int_{-\infty}^{\infty} (\mathrm{u}_1^2 + 2u_\alpha \mathrm{u}_2)\, d\xi = \mathrm{h}_2^{(s)} + \mathrm{h}_2^{(n)} \tag{27}$$

has both a singular part $\mathrm{h}_2^{(s)}$ and a non-singular part $\mathrm{h}_2^{(n)}$. The non-singular part can be shown to equal zero, and I concentrate on the singular part,

$$\begin{aligned} \mathrm{h}_2^{(s)} = \lim_{\xi\to\infty} 16 \int_{-\infty}^{\infty} \frac{dk_1}{2\pi}\, r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi}\, r(k_2) \exp[i(k_1+k_2)q_\alpha] \\ \int_{-\infty}^{\xi} d\xi_1 \exp[-2i(k_1+k_2)\xi_1] \, \frac{4k_1^2k_2^2 - k_1k_2(k_1^2-\kappa_\alpha^2)(k_2^2-\kappa_\alpha^2)}{(k_1+i\kappa_\alpha)^2(k_2+i\kappa_\alpha)^2}. \end{aligned} \tag{28}$$

Since the limit operator is outside the integral over k_1 the exchange of the integrals over k_1 and ξ_1 is legitimate. To evaluate Eq. (28), I first explicitly carry out the integral over ξ_1 assuming both k_1 and k_2 are in the upper half Bargmann strip. We then lower the contour over k_1, avoiding the pole as shown in Fig. 3. The continuous part of the integral vanishes, leaving only the pole contribution. I thus find

$$\mathrm{h}_2^{(s)} = 8 \int_{-\infty}^{\infty} \frac{dk_2}{2\pi}\, |r(k_2)|^2 k_2^2, \tag{29}$$

where $|r(k)|$ indicates the usual absolute value on the real k-axis and its analytic extension elsewhere. I conclude

$$H_1 = \frac{16}{3}\kappa_\alpha^3 \; + \; 8 \int_{-\infty}^{\infty} \frac{dk}{2\pi}\, k^2 |r(k)|^2, \tag{30}$$

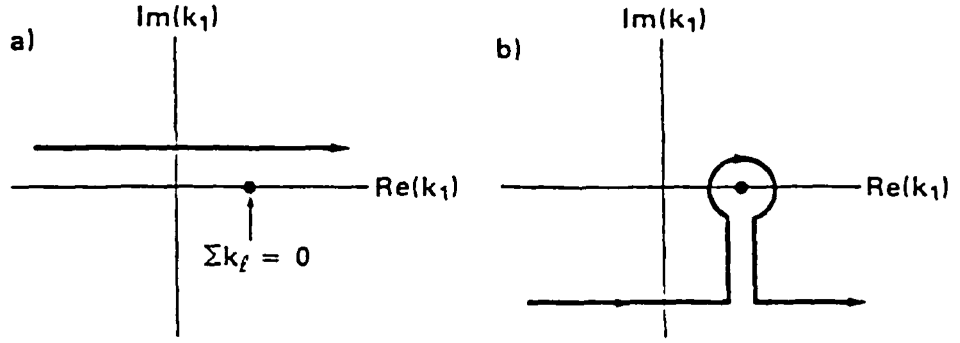


FIGURE 3. Illustration of the integration path as $Im(k_1)$ is decreased. The dot represents a pole. As $Im(k_1)$ is decreased, the pole yields a δ-function contribution.

which agrees through the order to which we are working with the exact result of Zakharov and Fadeev[19].

$$H_1 = \frac{16}{3}\kappa_\alpha^3 - 8\int_{-\infty}^{\infty} \frac{dk}{2\pi} k^2 \ln[1 - |r(k)|^2]. \tag{31}$$

Since H_1 only depends on the momenta, a soliton in the unperturbed equation remains a soliton, although its velocity of propagation changes.

When $p = 3$ or $p = 4$, the equations are still integrable, but the initial conditions for a soliton are changed, and, as a consequence, the h_n will depend on the coordinates just as in the non-integrable case where $p = 5$. Explicitly, I first obtain the result

$$\mathrm{h}_0 = \alpha^{(p)}\kappa_\alpha^{2p-1}, \tag{32}$$

where

$$\alpha^{(3)} = -\frac{128}{15}, \qquad \alpha^{(4)} = \frac{512}{35}, \qquad \alpha^{(5)} = -\frac{8192}{315}. \tag{33}$$

Next, I find

$$\mathrm{h}_1 = \beta^{(p)}\pi i \int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k^2 (k - i\kappa_\alpha)^2 P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(k\pi/\kappa_\alpha), \tag{34}$$

where

$$\beta^{(3)} = \frac{64}{3}, \qquad \beta^{(4)} = -\frac{256}{15}, \qquad \beta^{(5)} = \frac{512}{105}, \tag{35}$$

and

$$P^{(3)} = 1, \qquad P^{(4)} = (k^2 + 4\kappa_\alpha^2), \qquad P^{(5)} = (k^2 + 4\kappa_\alpha^2)(k^2 + 9\kappa_\alpha^2). \tag{36}$$

Finally, I obtain

$$\begin{aligned} \mathsf{h}_2 = \gamma^{(p)}\pi \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2) \frac{\exp[i(k_1+k_2)q_\alpha]}{(k_1+i\kappa_\alpha)^2(k_2+i\kappa_\alpha)^2} \\ Q^{(p)}(k_1,k_2,\kappa_\alpha)R^{(p)}(k_1,k_2,\kappa_\alpha)(k_1+k_2)\,\mathrm{csch}[\pi(k_1+k_2)/\kappa_\alpha], \end{aligned} \tag{37}$$

where

$$\gamma^{(3)} = -\frac{64}{3}, \qquad \gamma^{(4)} = \frac{256}{15}, \qquad \gamma^{(5)} = -\frac{512}{315}, \tag{38}$$

and

$$Q^{(3)} = 1, \quad Q^{(4)} = (k_1+k_2)^2+\kappa_\alpha^2, \quad Q^{(5)} = [(k_1+k_2)^2+\kappa_\alpha^2][(k_1+k_2)^2+4\kappa_\alpha^2]. \tag{39}$$

The quantity $R^{(p)}$ is a polynomial of the form

$$\begin{aligned} R^{(p)} = {} & a_1^{(p)}\kappa_\alpha^6 + a_2^{(p)}(k_1+k_2)^2\kappa_\alpha^4 + a_3^{(p)}k_1k_2\kappa_\alpha^4 + a_4^{(p)}(k_1+k_2)^4\kappa_\alpha^2 \\ & + a_5^{(p)}k_1k_2(k_1+k_2)^2\kappa_\alpha^2 + a_6^{(p)}k_1^2k_2^2\kappa_\alpha^2 + a_7^{(p)}(k_1+k_2)^6 \\ & + a_8^{(p)}k_1k_2(k_1+k_2)^4 + a_9^{(p)}k_1^2k_2^2(k_1+k_2)^2 + a_{10}^{(p)}k_1^3k_2^3, \end{aligned} \tag{40}$$

where

$$\begin{array}{lllll} a_1^{(3)} = 1 & a_2^{(3)} = 3 & a_3^{(3)} = -7 & a_4^{(3)} = 3 & a_5^{(3)} = -14 \\ a_1^{(4)} = 4 & a_2^{(4)} = 9 & a_3^{(4)} = -26 & a_4^{(4)} = 6 & a_5^{(4)} = -35 \\ a_1^{(5)} = 27 & a_2^{(5)} = 57 & a_3^{(5)} = -192 & a_4^{(5)} = 33 & a_5^{(5)} = -216 \\ a_6^{(3)} = 17 & a_7^{(3)} = 1 & a_8^{(3)} = -7 & a_9^{(3)} = 17 & a_{10}^{(3)} = -9 \\ a_6^{(4)} = 52 & a_7^{(4)} = 1 & a_8^{(4)} = -9 & a_9^{(4)} = 28 & a_{10}^{(4)} = -30 \\ a_6^{(5)} = 375 & a_7^{(5)} = 3 & a_8^{(5)} = -32 & a_9^{(5)} = 135 & a_{10}^{(5)} = -210 \end{array}$$

There are no singular contributions in any of these cases.

Structurally, these results are simpler than they perhaps appear at first glance. For all possible perturbations of the sort we are interested in, polynomial in u, its derivatives, and its integrals, one finds that h_n has the general form[11]

$$\begin{aligned} \mathsf{h}_n = \int_{-\infty}^{\infty} \frac{dk_1}{2\pi} |r(k_1)| \cdots \int_{-\infty}^{\infty} \frac{dk_n}{2\pi} |r(k_n)| \exp[-iq(k_1) \cdots - iq(k_2) \\ + i(k_1 + \cdots + k_n)q_\alpha] h_n(k_1, \cdots, k_n; \kappa_\alpha). \end{aligned} \tag{41}$$

Hence, the dependence on the canonical coordinates (as opposed to the canonical momenta) is entirely isolated inside the argument of imaginary exponentials. The quantity h_n depends only on κ_α and thus p_α. In general, it consists of a number of terms, each of which is a rational function of its arguments and may be multiplied by a δ-function factor of the form

$$\delta(\sum_j k_j)$$

due to a singular contribution containing two or more of the k_j. Those δ-functions which contain only two elements must be resolved explicitly since they have the effect of eliminating part of the coordinate dependence. Those δ-function factors containing three or more elements need not be resolved explicitly, although they can have an important effect on the behavior of the resonant denominators as I will describe shortly.

IV. LOWEST ORDER LIE GENERATOR AND RESONANT DENOMINATORS

The goal of Hamiltonian perturbation theory is to make a series of canonical transformations which eliminate the dependence of the Hamiltonian on the canonical coordinates through any given order. Through that order, the transformed action-angle variables evolve linearly in time,

$$\begin{aligned} J^{(n)} &= J_0^{(n)}, \\ \theta^{(n)} &= \theta_0^{(n)} + \Omega^{(n)} t, \end{aligned} \tag{42}$$

just as the original variables did in the unperturbed problem. Here the superscript n indicates the order of the transformation. The effect of these transformations is shown schematically in Fig. 4. Before the action-angle transformation, the original coordinates u evolve in a complicated way, shown as the dashed line to the left. However, after the action-angle transformation, the perturbed system *still* evolves in a complicated way. To obtain variables which evolve linearly through order n, we make further transformations using Hamiltonian perturbation theory. The evolution of these variables is shown schematically as the right-hand branch of Fig. 4. The three-sided path including this branch has through order n the same property that the original path of Fig. 1 has for the unperturbed system; it allows us to "win" over straightforward time integration when the time interval becomes sufficiently long.

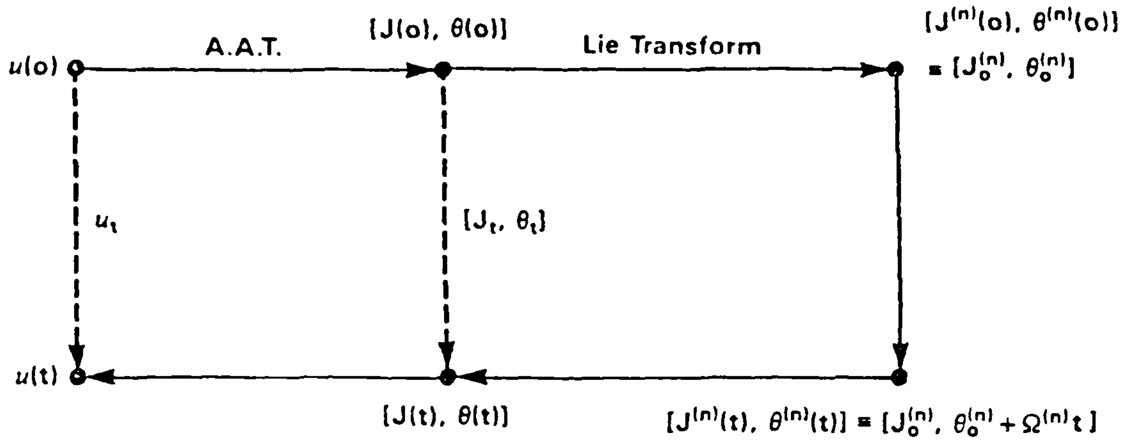

u(o)
A.A.T.
[J(o), θ(o)]
Lie Transform
[J^(n)(o), θ^(n)(o)] = [J_o^(n), θ_o^(n)]
u_t
[J_t, θ_t]
u(t)
[J(t), θ(t)]
[J^(n)(t), θ^(n)(t)] = [J_o^(n), θ_o^(n) + Ω^(n)t]

FIGURE 4. Schematic illustration of the way in which Hamiltonian perturbation theory can be used to solve perturbed equations. This figure should be compared with Fig. 1.

There is, however, a price to pay. These transformations generally diverge as one continues them to arbitrarily high order due to resonant or small denominators. The small denominators which appear in the theory equal zero in some cases on the real k-axis. To avoid this difficulty, I must extend integrals over k into the upper half Bargmann strip. To ensure that this extension is possible, we demand that $|u(x)| \to 0$ as $x \to \pm\infty$ faster than some exponential which implies that $r(k)$ is analytic in some strip surrounding the real k-axis. To ensure that all our integrals exist we impose the complementary constraint that $u(x)$ be analytic inside some strip around the real x-axis. Hence, $|r(k)|$ decreases faster than some exponential as $k \to \pm\infty$. If these conditions hold at any point in time, they hold for all times.

One of the beauties of the Hamiltonian approach is that it allows one to eliminate secularities in a simple, natural way. At any order n, we divide the Hamiltonian $H^{(n)}$ into two pieces

$$H^{(n)} = \hat{H}^{(n)} + \tilde{H}^{(n)}, \tag{43}$$

where the former is coordinate-independent and the latter is coordinate-dependent. We then eliminate $\tilde{H}^{(n)}$ to obtain $\tilde{H}^{(n+1)}$ which has a renormalized frequency $\Omega^{(n+1)}$ to which $\hat{H}^{(n)}$ contributes. At lowest order, for the examples which we are considering, this division is trivial. When $p = 2$, I find $\hat{H}_1 = H_1$ and $\tilde{H}_1 = 0$. As $\tilde{H}_1 = 0$, there is nothing to eliminate and no need to transform the Hamiltonian. When $p = 3$, 4, or 5, I find $\hat{H}_1 = \mathrm{h}_0$ and $\tilde{H}_1 = \mathrm{h}_1 + \mathrm{h}_2$ through second order in powers of

$r(k)$.

The Lie approach to Hamiltonian perturbation theory which I am using is based on two theorems[13,14]. I let $F[u], G[u]$, and $H[u]$ be arbitrary functionals of u. I also define the Lie operator $:F:$ corresponding to $F[u]$ as[17]

$$:F:G \equiv [F, G], \tag{44}$$

where $[F, G]$ indicates the Poisson bracket of F and G. The two theorems are:

1) *The transformation*

$$\bar{u} = \exp(:F:)u = \sum_{i=0}^{\infty} \frac{1}{i!}(:F:)^i u \tag{45}$$

is symplectic

2) *The relation*

$$\exp(-:F:)H[\exp(:F:)u] = H(u) \tag{46}$$

holds.

At lowest order, our task is to find a functional F_1 such that $H^{(1)} = \exp(-:F:)H$ no longer includes $\tilde{H}_1$. Then, from the second theorem, it follows that

$$H^{(1)}[p^{(1)}(k), q^{(1)}(k), p_\alpha^{(1)}, q_\alpha^{(1)}] = H[p(k), q(k), p_\alpha, q_\alpha], \tag{47}$$

while from the first theorem

$$\begin{aligned} p^{(1)}(k) &= \exp(\epsilon{:}F_1{:})p(k), \qquad & q^{(1)}(k) &= \exp(\epsilon{:}F_1{:})q(k), \\ p_\alpha^{(1)} &= \exp(\epsilon{:}F_1{:})p_\alpha, & q_\alpha^{(1)} &= \exp(\epsilon{:}F_1{:})q_\alpha, \end{aligned} \tag{48}$$

is a symplectic transformation and is just the transformation we want! The procedure is then continued to arbitrarily high order. Explicitly, we find through second order in ϵ

$$\begin{aligned} \exp(-\epsilon{:}F_1{:})H = H_0 + \epsilon\hat{H}_1 + \epsilon\tilde{H}_1 - \epsilon[F_1, H_0] \\ - \epsilon^2[F_1, \hat{H}_1] - \epsilon^2[F_1, \tilde{H}_1] + \frac{1}{2}\epsilon^2[F_1, [F_1, H_0]]. \end{aligned} \tag{49}$$

To eliminate $\tilde{H}_1$, we must set

$$\tilde{H}_1 = [F_1, H_0] = \left.\frac{dF_1}{dt}\right|_0. \tag{50}$$

In other words F_1 may be determined from $\tilde{H}_1$ by integration of the unperturbed orbits. Explicitly, I find

$$\int dt_0 \exp\left\{-i\sum_j [q(k_j) - k_j q_\alpha]\right\} = \frac{\exp\{-i\sum_j [q(k_j) - k_j q_\alpha]\}}{(-8i)\sum_j (k_j^3 + k_j \kappa_\alpha^2)}. \tag{51}$$

The zeroes in the small denominators

$$D = \sum_j (k_j^3 + k_j \kappa_\alpha^2), \tag{52}$$

may be avoided by integrating around them in the complex k-plane as needed.

For the cases of interest here, I find, expanding F_1 in powers of $r(k)$, that $F_1 = \mathrm{f}_1 + \mathrm{f}_2$, where f_1 and f_2 may be written

$$\mathrm{f}_1 = -\frac{\beta^{(p)}\pi}{8}\int_{-\infty}^{\infty} \frac{dk}{2\pi} r(k) \exp(ikq_\alpha) k \frac{k - i\kappa_\alpha}{k + i\kappa_\alpha} P^{(p)}(k, \kappa_\alpha) \operatorname{csch}(k\pi/\kappa_\alpha), \tag{53}$$

and

$$\mathrm{f}_2 = \frac{\gamma^{(p)}\pi i}{8}\int_{-\infty}^{\infty} \frac{dk_1}{2\pi} r(k_1) \int_{-\infty}^{\infty} \frac{dk_2}{2\pi} r(k_2)$$
$$\frac{\exp[i(k_1 + k_2)q_\alpha]}{(k_1 + i\kappa_\alpha)^2 (k_2 + i\kappa_\alpha)^2 [(k_1 + k_2)^2 - 3k_1 k_2 + \kappa_\alpha^2]}$$
$$Q^{(p)}(k_1, k_2, \kappa_\alpha) R^{(p)}(k_1, k_2, \kappa_\alpha) \operatorname{csch}[\pi(k_1 + k_2)/\kappa_\alpha]. \tag{54}$$

It is not difficult to show that the k-integrals in the Eqs. (53) and (54) are well-defined when both k_1 and k_2 are in the upper half Bargmann strip. More generally, if we consider the solution to the expression $D = 0$, I find that as long as at least two of the k_j are not tied together by a single δ-funtion factor and I assume that all the k_j except one which I designate k_l, are arbitrarily close to being purely real, then $D = 0$ is possible only if $Im(k_l) = 0$ or $Im(k_l) > \kappa_\alpha$. By choosing $Im(k_l) = \kappa_\alpha/2$, it is possible to bound D away from zero. If *all* the k_j are tied together by a δ-function factor then it is no longer possible to bound D away from zero, although I can always avoid having it equal zero by an appropriate choice of the k-integration contour. As a consequence of these latter terms, the perturbation theory is expected to ultimately diverge, although it is finite order-by-order. Physically, these terms correspond to a radiation component which travels with the solitary wave and only slowly disappears as $t \to +\infty$. More details on the resonant denominators may be found in reference 11.

V. CALCULATION OF THE HIGHER ORDER HAMILTONIAN

Having determined F_1, I may now calculate $H^{(1)}$, the transformed Hamiltonian. From Eqs. (49) and (50), it follows that

$$H^{(1)} = \exp(-\epsilon{:}\,F_1{:}\,)H = H_0 + \epsilon\hat{H}_1 - \epsilon^2[F_1, \hat{H}_1] - \frac{\epsilon^2}{2}[F_1, \tilde{H}_1]. \tag{55}$$

Noting that

$$\frac{\partial|r(k)|}{\partial p(k)} = \frac{\pi}{4k}\frac{1-|r(k)|^2}{|r(k)|}, \tag{56}$$

I find that in order to keep terms through order ϵ^2 in the equations of motion, where I assume that $|r(k)|$ is of order ϵ, I must keep terms in the perturbed Hamiltonian of order $\epsilon^2|r(k)|$ and $\epsilon|r(k)|^2$ as well as terms of order ϵ^2 and $\epsilon|r(k)|$.

The calculation of $H^{(1)}$ for the examples which I am considering here is straightforward, albeit somewhat lengthy. I will concentrate here on calculating in detail the term which contributes to $\hat{H}^{(1)}$ at order ϵ^2 in the case $p = 3$ and simply record the full results. The term in Eq. (55) on which we concentrate is $[\mathrm{f}_1, \mathrm{h}_1]$ which is part of $[F_1, H_1]$. Only the continuous portion of this Poisson bracket contributes since the soliton portion yields a term of order $\epsilon^2[r(k)]^2$. We first find that when $p = 3$

$$\begin{aligned}\frac{\partial \mathrm{f}_1}{\partial p(k)} = &-\frac{\pi}{3}\frac{1-|r(k)|^2}{|r(k)|}\exp[-iq(k)+ikq_\alpha]\frac{k-i\kappa_\alpha}{k+i\kappa_\alpha}\operatorname{csch}(\pi k/\kappa_\alpha)\\ &-\frac{\pi}{3}\frac{1-|r(k)|^2}{|r(k)|}\exp[iq(k)-ikq_\alpha]\frac{k+i\kappa_\alpha}{k-i\kappa_\alpha}\operatorname{csch}(\pi k/\kappa_\alpha),\end{aligned} \tag{57}$$

and

$$\begin{aligned}\frac{\partial \mathrm{f}_1}{\partial q(k)} = &\frac{4i}{3}|r(k)|\exp[-iq(k)+ikq_\alpha]k\frac{k-i\kappa_\alpha}{k+i\kappa_\alpha}\operatorname{csch}(\pi k/\kappa_\alpha)\\ &-\frac{4i}{3}|r(k)|\exp[iq(k)-ikq_\alpha]k\frac{k+i\kappa_\alpha}{k-i\kappa_\alpha}\operatorname{csch}(\pi k/\kappa_\alpha),\end{aligned} \tag{58}$$

where k is real and I have used the relations $|r(-k)| = |r(k)|, q(-k) = -q(k)$. Similar results can be obtained for $\partial\mathrm{h}_1/\partial p(k)$ and $\partial\mathrm{h}_1/\partial q(k)$. The operators $\partial/\partial q(k)$ and

$\partial/\partial q(k)$ are anti-symmetric in k; hence, the k-integrals can always be extended from the half interval $[0,\infty)$ to the full interval $(-\infty,\infty)$ and from there into the upper half Bargmann strip. In the case considered here I find through the order to which I am working,

$$\begin{aligned}
[\mathrm{f}_1,\mathrm{h}_1] &= \int_0^\infty dk\left(\frac{\partial \mathrm{f}_1}{\partial q(k)}\frac{\partial \mathrm{h}_1}{\partial p(k)} - \frac{\partial \mathrm{f}_1}{\partial p(k)}\frac{\partial \mathrm{h}_1}{\partial q(k)}\right)\\
&= -\frac{128\pi^2}{9}\int_\infty^\infty \frac{dk}{2\pi}\,[1-|r(k)|^2]k^2(k^2+\kappa_\alpha^2)\,\mathrm{csch}^2(\pi k/\kappa_\alpha)\\
&= -\frac{128\pi^2}{9}\int_{-\infty}^\infty \frac{dk}{2\pi}\,k^2(k^2+\kappa_\alpha^2)\,\mathrm{csch}^2(\pi k/\kappa_\alpha)\\
&= -\frac{128}{45}\kappa_\alpha^5.
\end{aligned} \tag{59}$$

In general, I may write

$$H^{(1)} = H_0 + \epsilon\hat{H}_1 + \epsilon^2\hat{H}_2 + \epsilon^2\tilde{H}_2. \tag{60}$$

I now find

$$\hat{H}_2 = \overline{\alpha}^{(p)}\kappa_\alpha^{4p-7},$$

where

$$\overline{\alpha}^{(3)} = -\frac{128}{45}, \qquad \overline{\alpha}^{(4)} = \frac{53,248}{1575}, \qquad \overline{\alpha}^{(5)} = \frac{128,712,704}{525,525}, \tag{61}$$

and

$$\begin{aligned}
\tilde{H}_2 = {}&\overline{\beta}^{(p)}\pi i\int_{-\infty}^\infty \frac{dk}{2\pi}\,r(k)\exp(ikq_\alpha)k^2\kappa_\alpha^{p-2}\frac{k-i\kappa_\alpha}{k+i\kappa_\alpha}P^{(p)}(k,\kappa_\alpha)\,\mathrm{csch}(\pi k/\kappa_\alpha)\\
&+\overline{\gamma}^{(p)}\pi^2 i\int_{-\infty}^\infty \frac{dk_1}{2\pi}\int_{-\infty}^\infty \frac{dk_2}{2\pi}\,r(k_2)\exp(ik_2q_\alpha)\frac{\overline{Q}^{(p)}(k_1,k_2,\kappa_\alpha)}{(k_2+i\kappa_\alpha)^2}\\
&R^{(p)}(k_1,k_2,\kappa_\alpha)\left[\frac{k_1}{(k_1+k_2)^2-3k_1k_2+\kappa_\alpha^2}+\frac{(k_1+k_2)}{(k_1^2+\kappa_\alpha^2)}\right]\\
&\mathrm{csch}(\pi k_1/\kappa_\alpha)\,\mathrm{csch}[\pi(k_1+k_2)/\kappa_\alpha)].
\end{aligned} \tag{62}$$

The quantities $P^{(p)}$ and $R^{(p)}$ are defined in Eqs. (36) and (40) respectively. The factors $\overline{\beta}^{(p)}$ and $\overline{\gamma}^{(p)}$ equal

$$\begin{aligned}
\overline{\beta}^{(3)} &= -\frac{128}{9}, \qquad \overline{\beta}^{(4)} = -\frac{2048}{75}, \qquad \overline{\beta}^{(5)} = -\frac{65,536}{3675},\\
\overline{\gamma}^{(3)} &= -\frac{128}{9}, \qquad \overline{\gamma}^{(4)} = -\frac{2048}{225}, \qquad \overline{\gamma}^{(5)} = -\frac{8192}{33,075},
\end{aligned} \tag{63}$$

and the $\overline{Q}^{(r)}$ equal

$$\begin{aligned}
\overline{Q}^{(3)} &= 1, \\
\overline{Q}^{(4)} &= [(k_1+k_2)^2+\kappa_\alpha^2](k_1^2+4\kappa_\alpha^2), \\
\overline{Q}^{(5)} &= [(k_1+k_2)^2+\kappa_\alpha^2][(k_1+k_2)^2+4\kappa_\alpha^2](k_1^2+4\kappa_\alpha^2)(k_1^2+9\kappa_\alpha^2).
\end{aligned} \tag{64}$$

Given the explicit form for $H^{(1)}$, we may now go on to calculate F_2 and $H^{(2)}$. We find explicitly

$$\begin{aligned}
F_2 = &-\frac{\overline{\beta}^{(p)}\pi}{8}\int_{-\infty}^{\infty}\frac{dk}{2\pi}\,r(k)\exp(ikq_\alpha)k\kappa_\alpha^{2p-2}\frac{1}{(k+i\kappa_\alpha)^2}P^{(p)}(k,\kappa_\alpha)\,\mathrm{csch}(\pi k/ka) \\
&-\frac{\overline{\gamma}^{(p)}\pi}{8}\int_{-\infty}^{\infty}\frac{dk_1}{2\pi}\int_{-\infty}^{\infty}\frac{dk_2}{2\pi}\,r(k_2)\exp(ik_2q_\alpha)\frac{\overline{Q}^{(p)}(k_1,k_2,\kappa_\alpha)}{k_2(k_1^2+\kappa_\alpha^2)(k_2+i\kappa_\alpha)^2} \\
&R^{(p)}(k_1,k_2,\kappa_\alpha)\left[\frac{k_1}{(k_1+k_2)^2-3k_1k_2+\kappa_\alpha^2}+\frac{(k_1+k_2)}{(k_1^2+\kappa_\alpha^2)}\right],
\end{aligned} \tag{65}$$

and

$$H^{(2)} = H_0+\epsilon\hat{H}_1+\epsilon^2\hat{H}_2. \tag{66}$$

Through the order to which we are working the Hamiltonian depends only on the canonical momenta. One can directly verify that all the k-integrals in $\tilde{H}_2$ and F_2 are well-defined when carried out in the upper half Bargmann strip.

VI. CALCULATION OF THE FIRST ORDER POTENTIAL

Having determined F_1 and F_2, we can in principle, calculate the second order potential through the formula

$$u^{(2)} = \exp(-\epsilon^2{:}F_2{:})\exp(-\epsilon{:}F_1{:})u; \tag{67}$$

however, I restrict myself now to calculating $u^{(1)} = \exp(-\epsilon{:}F_1{:})u$ in order to keep the algebra within reasonable bounds. Having determined $u^{(1)}$, I can check the Hamiltonian approach by finding the first order solitary wave solution and comparing the results to what is obtained using simpler methods which do not apply to arbitrary initial conditions. Such a check has already been carried out for other simple examples[10]. Calculating $u^{(1)}$ when $p = 3$ and setting $r^{(1)}(k) = 0$, I obtain for the solitary wave solution

$$u_s = -2\kappa_\alpha^2 \operatorname{sech}^2(\kappa_\alpha \xi) - \frac{2}{3}\epsilon\kappa_\alpha^2\Big[\operatorname{sech}^2(\kappa_\alpha\xi) \\ - 2(1 + 2\kappa_\alpha\xi)\operatorname{sech}^2(\kappa_\alpha\xi)\tanh(\kappa_\alpha\xi)\Big], \qquad (68)$$

where I have left out the superscript 1 which should be on κ_α and ξ. I note that the result is non-uniform in ξ although it does fall off faster than some exponential as $|\xi| \to \pm\infty$. From the Hamiltonian, I find

$$\kappa_\alpha\dot{\xi} = \kappa_\alpha\dot{q}_\alpha/2 = -4\kappa_\alpha^3 - \frac{16}{3}\epsilon\kappa_\alpha^2, \qquad (69)$$

Combining Eqs. (68) and (69), I find that through the order to which I am working

$$u_s = -2\kappa_\alpha^2(1 + \frac{1}{3}\epsilon)\operatorname{sech}^2\left[\kappa_\alpha(1 + \frac{2}{3}\epsilon)\xi + \frac{\epsilon}{3}\right] \\ = -2\kappa_\alpha^2(1 + \frac{1}{3}\epsilon)\operatorname{sech}^2\left[\kappa_\alpha(1 + \frac{2}{3}\epsilon)\xi_0 - 4\kappa_\alpha^3(1 + \frac{2\epsilon}{3})(1 + \frac{4\epsilon}{3})t + \frac{\epsilon}{3}\right] \qquad (70)$$

where ξ_0 is a constant of integration. Letting $\tilde{\kappa} = \kappa_\alpha(1 + 2\epsilon/3)$, we conclude

$$u_s = -2\tilde{\kappa}^2(1 - \epsilon)\operatorname{sech}^2\left[\tilde{\kappa}\xi_0 - 4\tilde{\kappa}^3 t + \frac{\epsilon}{3}\right], \qquad (71)$$

which is the same as the exact solution

$$u_s = -\frac{2\tilde{\kappa}^2}{1 + \epsilon}\operatorname{sech}^2\left[\tilde{\kappa}\xi_0 - 4\tilde{\kappa}^3 t + \frac{\epsilon}{3}\right], \qquad (72)$$

through the order to which we are working.

I have obtained similar results for the cases $p = 4$ and $p = 5$. In the former case I compared the results of my theory to what is obtained from the exact solution. In the latter case, no exact solution exists, and I compared the results with what is obtained from the expansion procedure of Kodama and Taniuti[20]. I have also explicitly verified that all the k-integrals which appear in $u^{(2)}$ are finite, although I have not carried them out in detail.

VII. CONCLUSIONS AND ACKNOWLEDGMENTS

In past work, I have used Hamiltonian perturbation methods to show that solitary waves emerge "to all orders" in a small parameter from arbitrary initial data. In this work, I apply the results to a second order calculation of some simple examples, $H_1 = u^r$. I have shown explicitly how to eliminate secularities by splitting the Hamiltonian into its coordinate-dependent and coordinate-independent pieces. I have also calculated the first order potentials and, from that, extracted the solitary wave structure. The results agree with previous theoretical work.

This work was supported by the Naval Research Laboratory and Science Applications International Corporation.

REFERENCES

[1] D.J. Korteweg and G. DeVries, "On the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves," Philos. Mag. Ser. 5 39, 422–443 (1895).

[2] H. Washimi and T. Taniuti, "Propagation of ion acoustic solitary waves of small amplitude," Phys. Rev. Lett. **17**, 996–998 (1966).

[3] C.R. Menyuk and H.H. Chen, "Hamiltonian Structure of the higher-order corrections to the Korteweg-de Vries equation," Phys. Rev. Lett. **55**, 1809–1811 (1985).

[4] C.R. Menyuk and H.H. Chen, "On the Hamiltonian structure of ion-acoustic plasma waves and water waves in channels," Phys. Fluids **29**, 998–1003 (1986).

[5] J.L. Hammack and H. Segur, "The Korteweg-de Vries equation and water waves, Part 2. Comparison with experiments," J. Fluid Mech. **65**, 289–314 (1974).

[6] P.D. Weidman and T. Maxworthy, "Experiments on strong interaction between solitary waves," J. Fluid Mech. **85**, 417–431 (1978).

[7] H. Ikezi, "Experiments on ion acoustic solitary waves," Phys. Fluids **16**, 1668–1675 (1973).

[8] S. Watanabe, "Ion-acoustic solitons excited by a single grid," **14**, 353–364 (1975).

[9] C.R. Menyuk, "Lie perturbation theory for an infinite-dimensional Hamiltonian system," in *Proceedings of the 13th International Colloquium on Group Theoretical Methods in Physics*, edited by W.W. Zachary (World Scientific Publ., Singapore, 1984), pp. 54–57.

[10] C.R. Menyuk, "Origin of solitons in the 'real' world," Phys. Rev. A **33**, 4367–4374 (1986).

[11] C.R. Menyuk, "Hamiltonian perturbations of the Korteweg-de Vries equation," (to appear).

[12] C.R. Menyuk, "Finite time solution of the perturbed Korteweg-de Vries equation," (to appear).

[13] G.-I. Hori, "Theory of general perturbations with unspecified canonical variables," Publ. Astron. Soc. Japan **18**, 287–296 (1966).

[14] A. Deprit, "Canonical transformations depending on a small parameter," Cel. Mech. **1**, 12–30 (1969).

[15] V.I. Karpman and E.M. Maslov, "Perturbation theory for solitons," Zh.E.T.F. **73**, 537–559 (1977) [Sov. Phys. JETP **46**, 281–291 (1977)].

[16] D.J. Kaup and A.C. Newell, "Solitons as particles, oscillators, and in slowly changing media: A singular perturbation theory," Proc. Roy. Soc. London A, **361**, 413–446 (1978).

[17] A.J. Dragt, "Lectures on nonlinear orbit dynamics," *in Physics of High Energy Particle Accelerators*, edited by R.A. Carrigan, F.R. Huson, and M. Month (A.I.P. Conf. Proc. no. 87, New York, 1982), pp. 147–313.

[18] M. Kruskal, "Asymptotic theory of Hamiltonian and other systems with all solutions nearly periodic," J. Math. Phys. **3**, 806–828 (1962).

[19] V.E. Zakharov and L.D. Fadeev, "Korteweg-de Vries equation: A completely integrable Hamiltonian system," Funct. Anal. Pril. [Funct. Anal. Appl. **5**, 280 (1971)].

[20] Y. Kodama and T. Taniuti, "Higher order approximation in the reductive perturbation method, I, the weakly dispersive system," J. Phys. Soc. Japan **45**, 298–310 (1978).

"Solitons in a birefringent Kerr medium," (C.R. Menyuk), in *Solitons and Chaos in Optical Systems*, H.C. Morris and D. Heffernan, eds. (World Scientific Publ., Singapore), to appear.

Solitons in a Birefringent Kerr Medium

C.R. Menyuk*
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228
USA

ABSTRACT

Equations are derived which govern wave propagation in a birefringent Kerr medium. Painlevé analysis indicates that these equations are integrable when the two polarizations are uncoupled or when the Kerr coefficient for each polarization depends on the total intensity. In the latter case, the equation's integrability was first proved by Manakov who found single soliton solutions. Here, the single soliton solutions that he found are extended.

I. INTRODUCTION

Many optical media are birefringent, and, as a consequence of this birefringence, have two normal modes with preferred axes of propagation. If the modes are linearly polarized, then we may designate the axes $\hat{\mathbf{e}}_x$ and $\hat{\mathbf{e}}_y$ which correspond to two orthogonal, real directions; however, if the modes are circularly polarized, then we designate the axes $\hat{\mathbf{e}}_r = (\hat{\mathbf{e}}_x - i\hat{\mathbf{e}}_y)/\sqrt{2}$ and $\hat{\mathbf{e}}_l = (\hat{\mathbf{e}}_x + i\hat{\mathbf{e}}_y)/\sqrt{2}$ which are no longer real.

If the nonlinear dielectric medium can be considered isotropic, then the lowest order nonlinear interaction which will appear is the cubic or Kerr nonlinearity [1–4]. In an intermediate range of birefringence, to be defined more precisely later in this paper, we then find

$$\begin{aligned} iu_\xi + \frac{1}{2}u_{ss} + \left(|u|^2 + B|v|^2\right)u &= 0, \\ iv_\xi + \frac{1}{2}v_{ss} + \left(B|u|^2 + |v|^2\right)v &= 0, \end{aligned} \tag{1}$$

where u and v represent the amplitude envelopes of two normal modes, ξ and s are normalized distance along the medium, and B is a parameter whose value depends on the details of the nonlinear dielectric response, although it is always $O(1)$. The most important single case is when the nonlinear dielectric response can be considered instantaneous, as is the case is optical fibers [5]. One then finds $B = 2/3$.

We note that while the coefficient of birefringence does not appear explicitly in Eq. (1), the transformation

$$u' = u\cos\theta + v\sin\theta,$$
$$v' = v\cos\theta - u\sin\theta, \tag{2}$$

does not leave Eq. (1) invariant unless $B = 1$. This invariance is a fundamental symmetry requirement if the normal modes are linearly polarized. Hence, the birefringence serves to break the azimuthal symmetry in this case.

Recently, Eq. (1) has beens subjected to intensive study due to the interest in optical fiber applications [6–9]. Unfortunately, these equations appear to be non-integrable when $B = 2/3$. Still, as Manakov showed some time ago, Eq. (1) is integrable when $B = 1$. Eq. (1) is also integrable when $B = 0$ since Eq. (1) reduces to two uncoupled nonlinear Schrödinger equations. We have carried out a Painlevé analysis which indicates that these are the only integrable cases.

The case $B = 1$, aside from its intrinsic interest, is a useful starting point for studying more general B-values. In his paper, Manakov [10] showed how to solve the initial value problem and extracted those single soliton solutions where u and v are both proportional to $\mathrm{sech}(\alpha s)$. We find more general single soliton solutions by a direct search for stationary solutions of Eq. (1). These solutions can be obtained by using a procedure first described by Darboux [11] and based on the original work of Bertrand [12] and Liouville [13].

In Sec. II of this paper, we give a brief derivation of Eq. (1). Our goal here is not rigor, but rather to elucidate what we consider to be the most important physical points. In Sec. III, we outline the Painlevé analysis and show how to obtain single soliton solutions of Eq. (1). The conclusions are in Sec. IV.

II. THE BASIC EQUATION

Recently, there have been several derivations of the nonlinear Schrödinger equation for applications to optical fibers and other optical systems. (See, *e.g.*, [3, 4, 14–16]). We shall present a simple derivation which can easily be made more rigorous by following the approach of [16]. We consider one-dimensional propagation in a homogeneous medium and ignore transverse effects.

In the slowly varying envelope approximation, we may assume that the E-field has the form

$$\mathbf{E}(z,t) = \mathbf{E}^{+}(z,t) + \mathbf{E}^{-}(z,t), \tag{3}$$

where $\mathbf{E}$ is the real field, $\mathbf{E}^+$ is the contribution to $\mathbf{E}$ near the carrier frequency $\omega = \omega_0$, and $\mathbf{E}^-$ is the contribution to $\mathbf{E}$ near $\omega = -\omega_0$. The Fourier transform of $\mathbf{E}$ is zero outside a small range of frequencies surrounding $\omega = \omega_0$ and $\omega = -\omega_0$. Since $\mathbf{E}(z,t)$ is real, it immediately follows that $\tilde{\mathbf{E}}(z,\omega)$, the Fourier transform of $\mathbf{E}(z,t)$, satisfies the relation $\tilde{\mathbf{E}}(z,-\omega) = \tilde{\mathbf{E}}^*(z,\omega)$ from which we conclude $\tilde{\mathbf{E}}^-(z,-\omega) = \tilde{\mathbf{E}}^{+*}(z,\omega)$. For each normal mode of the medium, we may now write

$$\begin{aligned} \mathbf{P}_1(z,t) &= \frac{1}{2\pi}\int_{-\infty}^{t} \chi_1(t-t')\mathbf{E}_1(z,t')\,dt', \\ \mathbf{P}_2(z,t) &= \frac{1}{2\pi}\int_{-\infty}^{t} \chi_2(t-t')\mathbf{E}_2(z,t')\,dt', \end{aligned} \tag{4}$$

where $\mathbf{P}_1$ and $\mathbf{P}_2$ indicate the linear polarizabilities in each component of the wave. Writing the Fourier transform

$$\tilde{A}(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} A(t)\exp(i\omega t)\,dt, \tag{5}$$

we find

$$\tilde{\mathbf{P}}_{1,2}(z,\omega) = \tilde{\chi}_{1,2}(z,\omega)\tilde{\mathbf{E}}_{1,2}(z,\omega), \tag{6}$$

or, separating out the + and − contributions [2],

$$\begin{aligned} \tilde{\mathbf{P}}_{1,2}(z,\omega) &= \tilde{P}^+_{1,2}(z,\omega)\hat{\mathbf{e}}_{1,2}(\omega) + \tilde{P}^-_{1,2}(z,\omega)\hat{\mathbf{e}}^*_{1,2}(\omega), \\ \tilde{\mathbf{E}}_{1,2}(z,\omega) &= \tilde{E}^+_{1,2}(z,\omega)\hat{\mathbf{e}}_{1,2}(\omega) + \tilde{E}^-_{1,2}(z,\omega)\hat{\mathbf{e}}^*_{1,2}(\omega), \end{aligned} \tag{7}$$

where the unit vectors satisfy the relations

$$\hat{\mathbf{e}}_1\cdot\hat{\mathbf{e}}_1^* = \hat{\mathbf{e}}_2\cdot\hat{\mathbf{e}}_2^* = 1, \qquad \hat{\mathbf{e}}_1\cdot\hat{\mathbf{e}}_2^* = \hat{\mathbf{e}}_2\cdot\hat{\mathbf{e}}_1^* = 0. \tag{8}$$

We concentrate on $\tilde{P}_1^+$. Since the relation $\tilde{P}_1^- = \tilde{P}_1^{+*}$ holds, knowledge of $\tilde{P}_1^+$ is sufficient to determine $\tilde{P}_1^-$; $\tilde{P}_2^+$ can then be determined by analogy with $\tilde{P}_1^+$. It is useful to consider the slowly varying envelopes of the polarizability and the field,

$$\begin{aligned} \rho(z,t) &= P_1^+(z,t)\exp(-ik_0 z + i\omega_0 t), \\ U(z,t) &= E_1^+(z,t)\exp(-ik_0 z + i\omega_0 t), \end{aligned} \tag{9}$$

where we will specify k_0 shortly. From Eqs. (8) and (9), we find

$$\rho(z,t) = \int_{-\infty}^{\infty} \tilde{\chi}(\omega+\omega_0)\tilde{U}(z,\omega)\exp(-i\omega t)\,d\omega. \tag{10}$$

The quantity $\tilde{U}(z,\omega)$ is peaked in a small region surrounding $\omega = 0$, and we assume that $\tilde{\chi}$ is slowly varying throughout this region. Thus, we may write

$$\tilde{\chi}_1(\omega+\omega_0) \simeq \tilde{\chi}_1(\omega_0) + \tilde{\chi}_1'(\omega_0)\omega + \frac{1}{2}\tilde{\chi}_1''(\omega_0)\omega^2, \tag{11}$$

where $\tilde{\chi}_1'(\omega_0)$ and $\tilde{\chi}_1''(\omega_0)$ are the first and second derivatives of $\tilde{\chi}_1(\omega+\omega_0)$ evaluated at $\omega = 0$, leading to the result

$$\rho = \tilde{\chi}_1(\omega_0)u + i\tilde{\chi}_1'(\omega_0)\frac{\partial u}{\partial t} - \frac{1}{2}\tilde{\chi}_1''(\omega_0)\frac{\partial^2 u}{\partial t^2}. \tag{12}$$

We now recall

$$\tilde{D}_1(z,\omega) = \left[1+\tilde{\chi}_1(\omega)\right]\tilde{E}_1(z,\omega) = \tilde{\epsilon}_1(\omega)\tilde{E}_1(z,\omega), \tag{13}$$

where $\tilde{\epsilon}_1(\omega)$ is the dielectric response. We further recall, from Maxwell's equations,

$$\nabla^2\tilde{\mathbf{E}}_1 + \frac{\omega^2}{c^2}\tilde{\mathbf{D}}_1 = 0, \tag{14}$$

and we define

$$k(\omega) = \frac{\omega}{c}\left[\tilde{\epsilon}_1(\omega)\right]^{1/2}, \tag{15}$$

corresponding to positive propagation. Letting $k_0 \equiv k(\omega_0)$, we now find from Eqs. (9, 13–15),

$$i\frac{\partial U}{\partial z} + ik_0'\frac{\partial U}{\partial t} - \frac{1}{2}k_0''\frac{\partial^2 U}{\partial t^2} = 0, \tag{16}$$

where k_0' and k_0'' are the first and second derivatives of $k(\omega)$, evaluated at $\omega = \omega_0$. Similarly, we find

$$i\frac{\partial V}{\partial z} + il_0'\frac{\partial V}{\partial t} - \frac{1}{2}l_0''\frac{\partial^2 V}{\partial t^2} = 0, \tag{17}$$

where V is the envelope of E_2^+,

$$l(\omega) = \frac{\omega}{c}\left[\tilde{\epsilon}_2(\omega)\right]^{1/2} = \frac{\omega}{c}\left[1+\tilde{\chi}_2(\omega)\right]^{1/2}, \tag{18}$$

and l_0, l_0', and l_0'' are defined by analogy with k_0, k_0', and k_0''.

We now suppposed that the polarizability has a cubic component and that this cubic component is istropic. When both the anisotropy and nonlinearity are weak, the case of greatest practical interest, then anisotropy can be ignored in the nonlinear contribution at lowest order since the anisotropy is formally of higher order. The polarizability must have the form

$$\mathbf{P}(z,t) = \frac{1}{(2\pi)^3}\int_{-\infty}^{t} dt_1 \int_{-\infty}^{t} dt_2 \int_{-\infty}^{t} dt_3\, \chi(t-t_1, t-t_2; t-t_3) \left[\mathbf{E}(z,t_1)\cdot\mathbf{E}(z,t_2)\right]\mathbf{E}(z,t_3). \tag{19}$$

Equation (19) is the only cubic combination of E_x and E_y which is invariant under rotations and mirror reflections. From the form of Eq. (19), it follows that the dielectric function $\chi(\tau_1, \tau_2; \tau_3)$ is invariant under the interchange $\tau_1 \leftrightarrow \tau_2$ but not under the interchanges $\tau_1 \leftrightarrow \tau_3$ or $\tau_2 \leftrightarrow \tau_3$. We thus obtain

$$\begin{aligned} \mathbf{P}^+(z,t) = \frac{1}{(2\pi)^3} \int_{-\infty}^{t} dt_1 \int_{-\infty}^{t} dt_2 \int_{-\infty}^{t} dt_3\, \chi(t-t_1, t-t_2; t-t_3) \\ \{[2\mathbf{E}^+(z,t_1)\cdot\mathbf{E}^-(z,t_2)]\mathbf{E}^+(z,t_3) \\ + [\mathbf{E}^+(z,t_1)\cdot\mathbf{E}^+(z,t_2)]\mathbf{E}^-(z,t_3)\}. \end{aligned} \tag{20}$$

and a similar result for $\mathbf{P}^-$. The decomposition of Eq. (20) depends on the nature of the normal modes. For linearly polarized waves,

$$\begin{aligned} P_1^+(z,t) = \frac{1}{(2\pi)^3} \int_{-\infty}^{t} dt_1 \int_{-\infty}^{t} dt_2 \int_{-\infty}^{t} dt_3\, \chi(t-t_1, t-t_2; t-t_3) \\ \{2[E_1^+(z,t_1)E_1^-(z,t_2) + E_2^+(z,t_1)E_2^-(z,t_2)]E_1^+(z,t_3) \\ + [E_1^+(z,t_1)E_1^+(z,t_2) + E_2^+(z,t_1)E_2^+(z,t_2)]E_1^-(z,t_3)\}, \end{aligned} \tag{21}$$

with a similar result for P_2^+, while for circularly polarized waves

$$\begin{aligned} P_1^+(z,t) = \frac{1}{(2\pi)^3} \int_{-\infty}^{t} dt_1 \int_{-\infty}^{t} dt_2 \int_{-\infty}^{t} dt_3\, \chi(t-t_1, t-t_2; t-t_3) \\ \{2[E_1^+(z,t_1)E_1^-(z,t_1) + E_2^+(z,t_1)E_2^-(z,t_2)]E_1^+(z,t_3) \\ + 2[E_1^+(z,t_1)E_2^+(z,t_2)]E_2^-(z,t_3)\}. \end{aligned} \tag{22}$$

Making the slowly varying envelope approximation, just as in the linear case, and keeping only the lowest order terms in the expansion of $\tilde{\chi}$, we find in the case of linearly polarized waves that

$$\begin{aligned} \rho(z,t) = \alpha\{2(|U|^2 + |V|^2)\}U \\ + \beta\{U^2 + V^2 \exp[-2i(k_0 - l_0)z]\}U^*, \end{aligned} \tag{23}$$

where $\alpha = \tilde{\chi}(\omega_0, -\omega_0; \omega_0)$ and $\beta = \tilde{\chi}(\omega_0, \omega_0; -\omega_0)$. For circularly polarized modes, we obtain

$$\rho(z,t) = \alpha\{2(|U|^2 + |V|^2)\}U + 2\beta|V|^2U. \tag{24}$$

When the medium has an instantaneous response, $\tilde{\chi}(\omega_0, -\omega_0; \omega_0) = \tilde{\chi}(\omega_0, \omega_0; -\omega_0) = \tilde{\chi}(0,0;0)$, so that $\alpha = \beta$.

In many cases of practical interest, the birefringent beat length is short compared to the length scale of the pulse variation. Then, the term in Eq. (23) is rapidly oscillating and can be dropped. We now combine the effects of the linear and nonlinear polarizability. After transforming to the intermediate group velocity frame and appropriate normalization [8, 9], we find for linearly polarized waves,

$$\begin{aligned} iu_\xi + i\delta u_s + \frac{1}{2}u_{ss} + \left(|u|^2 + B|v|^2\right)u = 0, \\ iv_\xi - i\delta v_s + \frac{1}{2}v_{ss} + \left(B|u|^2 + |v|^2\right)v = 0, \end{aligned} \tag{25}$$

in the anomalous dispersion regime where $B = 2\alpha/(2\alpha+\beta)$. For circularly polarized waves, Eq. (25) still holds with $B = (\alpha+\beta)/\alpha$, and no assumption concerning the birefringence strength is required. The first derivatives in s can be removed by the transformation

$$\begin{aligned} \bar{u} = u \exp\left[i\delta(1 - \frac{\delta}{2})\xi - i\delta s\right], \\ \bar{v} = v \exp\left[-i\delta(1 + \frac{\delta}{2})\xi + i\delta s\right]. \end{aligned} \tag{26}$$

Removing the bars yields Eq. (1). We see that the Manakov equation results when $\beta = 0$.

It is worthy of note that when the birefringence is so weak that the exponential term in Eq. (1) can be set equal to 1, we find

$$\begin{aligned} iu_\xi + \frac{1}{2}u_{ss} + \left(|u|^2 + |v|^2\right)u + (1-B)\left(uv^* - vu^*\right)v = 0, \\ iv_\xi + \frac{1}{2}v_{ss} + \left(|u|^2 + |v|^2\right)v - (1-B)\left(uv^* - vu^*\right)v = 0. \end{aligned} \tag{27}$$

The final terms in Eq. (27) lead to ellipse rotation [1].

II. INTEGRABILITY AND SOLITONS

We now look for stationary solutions of Eq. (1) which have the form

$$\begin{aligned} u(\xi, s) = \exp(i\Omega_1\xi)f(s), \\ v(\xi, s) = \exp(i\Omega_2\xi)g(s), \end{aligned} \tag{28}$$

where f and g are real functions and Ω_1 and Ω_2 are two real parameters. In the case $B = 0$ where the single soliton solutions are well-known, we find that this *ansatz* yields the general solution to within a Galilean transformation. Substitution of Eq. (28) into Eq. (1) yields

$$\begin{aligned} f_{ss} - 2\Omega_1 f + 2(f^2 + Bg^2)f = 0, \\ g_{ss} - 2\Omega_2 g + 2(Bf^2 + g^2)g = 0. \end{aligned} \tag{29}$$

In the remainder of this section, we study Eq. (29). We apply Painlevé analysis [17] to Eq. (29) which indicates that it is only integrable when $B = 0$ or $B = 1$. Then, setting $B = 1$, we determine the homoclinic orbits which correspond to single soliton solutions.

A. Painlevé Analysis

Following the procedure of Ablowitz, *et al.* [17], we search for a Laurent series solution of Eq. (29),

$$\begin{aligned} f &= \sum_{j=0}^{\infty} a_j (s - s_0)^{p+j}, \\ g &= \sum_{j=0}^{\infty} b_j (s - s_0)^{q+j}, \end{aligned} \tag{30}$$

valid in the neighborhood of any singular point $s = s_0$. The only choice of p and q which allows us to balance leading terms in Eq. (29) while leading to four arbitrary coefficients in Eq. (30) is $p = q = -1$. We then find

$$a_0^2 = b_0^2 = -\frac{1}{B+1}. \tag{31}$$

We next determine the values of j at which arbitrary coefficients in the Laurent expansion enter. Letting $j = r$ designate these resonant values, we find that r satisfies the equations,

$$\begin{aligned} \left[(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1}\right] a_r &= \pm 4 \frac{B}{B+1} b_r, \\ \left[(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1}\right] b_r &= \pm 4 \frac{B}{B+1} a_r, \end{aligned} \tag{32}$$

where have made use of Eq. (31) to eliminate a_0 and b_0. We now find

$$(r-1)(r-2) - \frac{6}{B+1} - \frac{2B}{B+1} \mp \frac{4B}{B+1} = 0, \tag{33}$$

from which, taking the $-$ and $+$ signs in turn, we conclude that Eq. (32) has the roots

$$r = -1, \quad 4, \quad \frac{3}{2} \pm \frac{1}{2}\left(9 - 16\frac{B-1}{B+1}\right)^{1/2}. \tag{34}$$

The only values of B which yield real, integral roots are $B = 0$, in which case $r = -1$ and $r = 4$ are both double roots, or $B = 1$, in which case $r = -1, 0, 3$, and 4. Hence, the only values of B for which Eq. (29) can have the Painlevé property are $B = 0$ and $B = 1$.

To complete the Painlevé analysis, we must substitute Eq. (30) into Eq. (29) and show through $j = 4$ that no logarithmic singularities develop when $B = 0$ and $B = 1$. We have done so, but do not describe the algebraic details.

B. Soliton Solutions When $B = 1$

When $B = 1$, Eq. (29) is generated by the Hamiltonian

$$\mathcal{H} = \frac{1}{2}F^2 + \frac{1}{2}G^2 + \frac{1}{2}\left[(f^2+g^2)^2 - 2\Omega_1 f^2 - 2\Omega_2 g^2\right], \tag{35}$$

where $F = df/ds$ and $G = dg/ds$ are, respectively, the momenta canonical to f and g. The independent variable is s. A second, independent constant of the motion is

$$C = \frac{1}{2}(gF - fG)^2 + (\Omega_1 - \Omega_2)\left[F^2 - 2\Omega_1 f^2 + (f^2+g^2)f^2\right]. \tag{36}$$

Equation (36) implies the integrability of Eq. (29) when $B = 1$. When $\Omega_1 > 0$ and $\Omega_2 > 0$, homoclinic orbits exist which correspond to solitons. If $\Omega_1 = \Omega_2 \equiv \Omega$, then the solution

$$\begin{aligned} f(s) &= (2\Omega)^{1/2}\cos\alpha\,\mathrm{sech}\left[(2\Omega)^{1/2}s\right], \\ g(s) &= (2\Omega)^{1/2}\sin\alpha\,\mathrm{sech}\left[(2\Omega)^{1/2}s\right], \end{aligned} \tag{37}$$

corresponds to the solitons found by Manakov [10]. If $\Omega_1 \neq \Omega_2$, then the homoclinic *orbits are considerably more complicated.*

Some time ago, Darboux [11] shown that a two degree-of-freedom Hamiltonian system with a second integral quadratic in the momenta has a generic form. Once this form is obtained by using Bertrand's method [12], (see also [18]) the equations of motion can be reduced to quadratures using a procedure due to Liouville. To reduce our equation to this form, we first note that the potential contribution to $\mathcal{H}$ is

$$V(f,g) = \frac{1}{2}(f^2+g^2)^2 - \Omega_2(f^2+g^2) - (\Omega_1 - \Omega_2)f^2. \tag{38}$$

We next define new variables x and y such that

$$\begin{aligned} x^2 + y^2 &= f^2 + g^2 + \gamma, \\ x^2 - y^2 &= \left[(f^2+g^2+\gamma)^2 - 4\gamma f^2\right]^{1/2}, \end{aligned} \tag{39}$$

where $\gamma = 2(\Omega_1 - \Omega_2)$. The potential $V(x,y)$ now has the appropriate generic form,

$$V(x,y) = \frac{X(x) - Y(y)}{x^2 - y^2}, \tag{40}$$

where

$$X(\alpha) = Y(\alpha) \equiv A(\alpha) = \frac{1}{2}\alpha^2(\alpha^2 - 2\Omega_1)(\alpha^2 - 2\Omega_1 - 2\Omega_2). \tag{41}$$

To reduce the equations of motion to quadratures, we first write the kinetic contribution to $\mathcal{H}$,

$$T(F,G) = \frac{1}{2}(F^2 + G^2) = \frac{1}{2}(x^2 - y^2)\left(\frac{x_s^2}{x^2 - \gamma} + \frac{y_s^2}{\gamma - y^2}\right). \tag{42}$$

Defining now,

$$\tilde{x} = \int \frac{dx}{(x^2 - \gamma)^{1/2}} \qquad \text{and} \qquad \tilde{y} = \int \frac{dy}{(\gamma - y^2)^{1/2}}, \tag{43}$$

we note that T and V have the forms

$$\begin{aligned} T &= \frac{1}{2}\left[c_1(\tilde{x}) + c_2(\tilde{y})\right](\tilde{x}_s^2 + \tilde{y}_s^2), \\ V &= \frac{d_1(\tilde{x}) + d_2(\tilde{y})}{c_1(\tilde{x}) + c_2(\tilde{y})}. \end{aligned} \tag{44}$$

Defining further $c = c_1(\tilde{x}) + c_2(\tilde{y})$ and writing the Lagrangian

$$\frac{d}{ds}\left(\frac{\partial T}{\partial \tilde{x}_s}\right) - \frac{\partial T}{\partial \tilde{x}} = -\frac{\partial V}{\partial \tilde{x}}, \tag{45}$$

we obtain after some algebra

$$\frac{d}{ds}(c^2\tilde{x}_s^2) - c\tilde{x}_s\frac{\partial c}{\partial \tilde{x}_s}(\tilde{x}_s^2 + \tilde{y}_s^2) = -2c\tilde{x}_s\frac{\partial V}{\partial \tilde{x}}. \tag{46}$$

From the Hamiltonian, we find

$$\frac{1}{2}c(\tilde{x}_s^2 + \tilde{y}_s^2) = h - V, \tag{47}$$

where h is some constant, and, after some more algebra, we arrive at the expression

$$\frac{d}{ds}(c^2\tilde{x}_s^2) = 2\frac{d}{ds}(hc_1 - d_1), \tag{48}$$

or

$$\frac{1}{2}c^2\tilde{x}_s^2 = hc_1 - d_1 + \gamma_1, \tag{49}$$

where γ_1 is a constant of integration. Carrying out a similar operation for $\tilde{y}_s$, we finally conclude

$$(hc_1 - d_1 + \gamma_1)^{1/2}d\tilde{x} = (hc_2 - d_2 + \gamma_2)^{1/2}d\tilde{y}, \tag{50}$$

which reduces the problem to quadratures.

Closed form expressions can be found for the solitons and were recently reported by Cristodoulides and Joseph [19] with some generalization from the case considered here. We do not reproduce their analytic form since it is rather complicated; however, the physical structure of the solution is not difficult to determine. When $\Omega_1 > \Omega_2$, the f-component is sharper and the g-component dominates at large values of $|s|$. The self-similar structure retains its shape through a complex balance of the contributions of the two different components.

IV. CONCLUSIONS

In this paper, we have shown how the Kerr effect leads to the coupled nonlinear Schrödinger equation in a birefringent medium. Painlevé analysis indicates that these equations are only integrable in two special cases. In the first case, the two polarizations are uncoupled. In the second case, the nonlinar contribution of the two polarizations to the Kerr coefficient of each polarization is identical. This latter case was shown to be integrable by Manakov who found special single soliton solutions. We have extended his results by finding a more general class of single soliton solutions.

ACKNOWLEDGMENTS

The author is grateful to Dr. Alfred Ramani of the Ecole Polytechnique who first pointed out to him that Eq. (29) could be integrated using Darboux's approach. Early work on this subject was carried out at the Ecole Polytechnique with support from the Centre National d'Etudes en Télécommunications. Later work was supported by the Naval Research Laboratory.

REFERENCES

1. P. D. Maker and R. W. Terhune, "Study of optical effects due to an induced polarization third order in the electric field strength," Phys. Rev. **137**, A801–A818 (1965).
2. H. A. Haus, *Waves and Fields in Optoelectronics* (Prentice-Hall, Englewood Cliffs, NJ, 1984), pp. 358–359.
3. V. I. Talanov, "Self-focusing of wave beams in nonlinear media," Zh.E.T.F. Pis. Red. **2**, 218–223 (1965) [Sov. Phys. JETP Lett. **2**, 138–141 (1965)].
4. P. L. Kelley, "Self-focusing of optical beams," Phys. Rev. Lett. **15**, 1005–1008 (1965).
5. See, *e.g.*, R. H. Stolen, J. Botineau, and A. Ashkin, "Intensity discrimination of optical pulses with birefringent fibers," Opt. Lett. **7**, 512–514 (1982).

6. A. Hasegawa, "Self-confinement of multimode optical pulse in a glass fiber," Opt. Lett. **5**, 416–417 (1980).

7. B. Crosignani and P. Di Porto, "Soliton propagation in multimode optical fibers," Opt. Lett. **6**, 329–330 (1981).

8. C. R. Menyuk, "Nonlinear pulse propagation in birefringent optical fibers," IEEE J. Quantum Electron. **QE-23**, 174–176 (1987).

9. C. R. Menyuk, "Stability of solitons in birefringent optical fibers I: Equal propagation amplitudes," Opt. Lett. **12**, 614–616 (1987).

10. S. V. Manakov, "On the theory of two-dimensional stationary self-focusing of electromagnetic waves," Zh.E.T.F. **65**, 505–516 (1973) [Sov. Phys. JETP **38**, 248–253 (1974)].

11. G. Darboux, "Sur un problème de mécanique," Archives néerlandaises **6** (Ser. 2), 371–376 (1901). See also, E. T. Whittaker, *A Treatise on the Analytical Dynamics of Particles and Rigid Bodies* (Dover, New York, 1944), pp. 332–335.

12. M. J. Bertrand, "Sur les intégrales communes à plusieurs problèmes de mécanique," J. Math. Pures Appliquées **17** (Ser. 1), 121–174 (1852). See also, E. T. Whittaker, *op. cit.*, pp. 331–332.

13. J. Liouville, "Mémoire sur quelques cas particuliers où les équations différentielles du mouvement d'un point matériel peuvent s'intégrer," J. Math. Pures Appliquées **14** (Ser. 1), 257 (1849). See also, E. T. Whittaker, *op. cit.*, pp. 67–69.

14. A. Hasegawa and F. Tappert, "Transmission of stationary nonlinear optical pulses in dispersive dielectric fibers I. Anomalous dispersion," Appl. Phys. Lett. **23**, 142 (1973).

15. P. Bendow, P. D. Gianino, N. Tzoar, and M. Jain, "Theory of nonlinear pulse propagation in optical waveguides," J. Opt. Soc. Am. **70**, 539–546 (1980).

16. Y. Kodama, "Optical solitons in a monomode fiber," J. Stat. Phys. **39**, 597–614 (1985).

17. M. J. Ablowitz, A. Ramani, and H. Segur, "A connection between nonlinear evolution equations and ordinary differential equations of P-type. I," J. Math. Phys. **21**, 715–721 (1980).

18. B. Dorizzi, B. Grammaticos, and A. Ramani, "A new class of integrable systems," J. Math. Phys. **24**, 2282–2288 (1983).

19. D. N. Christodoulides and R. I. Joseph, "Vector solitons in birefringent nonlinear dispersive media," Opt. Lett. **13**, 53–55 (1988).

*Also at: Department of Electrical Engineering, University of Maryland, College Park, MD 21228

"Pump replication in stimulated Raman scattering using a crossed-beam geometry," (C.R. Menyuk, G. Hilfer, and J. Reintjes), in *Nonlinear Optics and Beam Coupling*, R.A. Fisher, ed. (SPIE Publ. no. 874, Bellingham, WA), to appear.

074 50

Pump replication in stimulated Raman scattering using a crossed-beam geometry

C. R. Menyuk* and G. Hilfer
Science Applications International Corporation, 1710 Goodridge Drive, McLean, Va. 22102

J. Reintjes
Laser Physics Branch, Naval Research Laboratory, Washington, DC 20375

ABSTRACT

A theory of side beam replication in a crossing-beam geometry is reported. It is shown that side beam replication is not expected to occur when the Fresnel number of the aberrations (FN_A) is large, while it is expected to occur when FN_A is small, in accord with experiments. An analytic threshold is derived for the value of FN_A at which side beam replication no longer occurs, and this threshold agrees well with the experiments. We propose a method for eliminating side beam replication at low values of FN_A.

1. INTRODUCTION

The theoretical and experimental work which has been carried out to date on Raman beam cleanup and beam combining of stationary waves has been strongly motivated by previous work on phase conjugation, mostly based on Brillouin scattering, rather than Raman scattering.[1,2] In both cases, four wave mixing processes are involved. In the early experiments of Goldhar and Murray[3] counter-propagating beams were considered and the effect of a finite pump beam correlation length was determined. They show that a large number of pump beams leads to averaging and a smoother Stokes output. Shortly thereafter, Chang and Djeu[4] carried out experiments in a co-propagating beam geometry. They found, in keeping with the theoretical predictions of Bespalov, *et al.*[5] that as the gain rose, the Stokes beam distortion increased, due to incomplete intensity averaging along the length of the amplifier. In later work, Goldhar, *et al.*[6] showed that their approach could be made more efficient by using a double-pass amplifier, and Chang, *et al.*[7,8] showed that far better output Stokes quality could be obtained if a multi-beam geometry with the central pump component removed, was used. More recently, Reintjes, *et al.*[9,10] have explored in considerable detail the different parameter regimes which occur in a multi-beam geometry and the behavior observed in the different regimes.

In their experiments, Reintjes, *et al.*[9,10] observed that the efficiency of beam cleanup is determined in large measure by the beam geometry and by the Fresnel number of the aberrations $FN_A = D_A^2/\lambda L$, where D_A is the transverse scale length of the aberrations, λ is the pump wavelength, and L is the interaction length. In a collinear beam geometry, with a large Fresnel number so that diffraction can be ignored, the same portions of the pump beam and Stokes beam continually interact. As a consequence, no intensity averaging can occur, and any amplitude structure in the pump will print through onto the Stokes, although no phase structure prints through. As the Fresnel number decreases, intensity averaging begins to occur, reducing the deleterious effect of amplitude aberrations. However, diffraction of phase structure into amplitude structure now occurs, so that some printing through of phase aberrations takes place. As the Fresnel number decreases yet further, the intensity averaging improves substantially, but it is always incomplete.

If we consider instead a multi-beam geometry where there is no on axis pump beam, shown schematically in Fig 1, then intensity averaging is considerably enhanced, and at small Fresnel numbers the Stokes beam is essentially diffraction limited. However, when the Fresnel number is not small, side beam replication can occur. That is to say, new Stokes beams can be created which propagate collinearly with the off-axis pump beams.

In this work, we theoretically examine the conditions under which side beam replication occurs. This replication is closely analogous to Brillouin phase conjugation due to four wave mixing, and we make heavy use of the approach which was first developped in theoretical studies of this effect.[1,2] Flusberg and Korff[11] have already noted this analogy, and they have made excellent use of it it in their recent study of Raman amplification in a collinear beam geometry. In the experiments of interest to us, however, this analogy is incomplete. The difference between k_L and k_S, the pump and Stokes wavenumbers, is quite large, amounting to 13% of k_L;[9,10] as a consequence, important modifications must be made in the theory.

Our theory leads us to propose a novel method for eliminating side beam replication without degrading pump beam quality by adjusting the phases of the incoming pump beams.

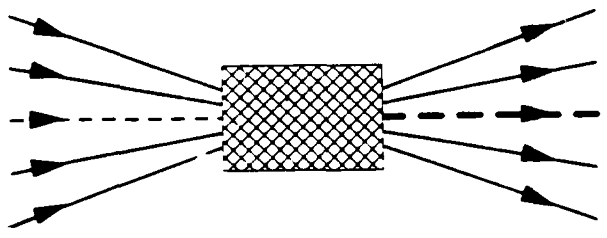


FIGURE 1 A schematic illustration of the crossing-beam geometry is shown The pump beams are shown as solid lines, and the Stokes beam is shown as a dashed line There is no pump beam propagating collinearly with the Stokes beam When the Stokes beam emerges from the interaction region, shown as a hatched box, it is amplified

014 30

2. THEORETICAL DEVELOPMENT

The basic equations which govern wave evolution in a Raman active medium are

$$\begin{aligned}\frac{\partial E_L}{\partial z} - \frac{i}{2k_L}\frac{\partial^2 E_L}{\partial y^2} &= -i\frac{k_L}{k_S}\kappa_2 Q E_S,\\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S}\frac{\partial^2 E_S}{\partial y^2} &= -i\kappa_2 Q^* E_L, \qquad (1)\\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i\kappa_1 E_S^* E_L,\end{aligned}$$

where E_L and E_S are the complex envelopes of the pump and Stokes waves, Q is the material excitation, κ_1 and κ_2 are the gain coefficients, and $\Gamma \equiv 1/T_2$ is the damping rate of the material excitation. Here, we consider only one transverse dimension for simplicity of presentation but note that the results which we will obtain hold without change for two transverse dimensions. To derive Eq. (1), we make a slowly varying envelope approximation, a paraxial wave approximation, and assume that the material excitation is far from saturation.

In the stationary limit of Eq. (1), where time dependent effects can be neglected, we find

$$Q = -i\kappa_1\frac{E_S^* E_L}{\Gamma}, \qquad (2)$$

and, as a consequence,

$$\begin{aligned}\frac{\partial E_L}{\partial z} - \frac{i}{2k_L}\frac{\partial^2 E_L}{\partial y^2} &= -\frac{k_L}{k_S}\frac{g}{2}|E_S|^2 E_L,\\ \frac{\partial E_S}{\partial z} - \frac{i}{2k_S}\frac{\partial^2 E_S}{\partial y^2} &= \frac{g}{2}|E_L|^2 E_S, \qquad (3)\end{aligned}$$

where $g = 2\kappa_1\kappa_2/\Gamma$. The experiments[9,10] indicate that the system's linear behavior (i.e., behavior when $|E_S| \ll |E_L|$) plays a crucial role in determining the quality of the emerging Stokes beam. We thus specialize to the linear limit where Eq. (3) becomes

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_L}\frac{\partial^2 E_L}{\partial y^2} = 0, \qquad (4a)$$

$$\frac{\partial E_L}{\partial z} - \frac{i}{2k_S}\frac{\partial^2 E_S}{\partial y^2} = \frac{g}{2}|E_L|^2 E_S. \qquad (4b)$$

In the multi-beam geometry that we are considering, it is always the case that in k_y-space, the Fourier transform space corresponding to the y-direction, the separation between the beams is much larger than the bandwidth of each individual beam. In other words, $(\Delta K)_{\rm sep} \gg (\Delta K)_{\rm beam}$, where $(\Delta K)_{\rm sep}$ is the minimum k_y-separation between the beams and $(\Delta K)_{\rm beam}$ is the maximum bandwidth of an individual beam. Given this condition, it is useful to decompose E_L and E_S into a sum of contributions from each beam in which the central wavenumber K_l of each beam is explicitly accounted for,

$$\begin{aligned}E_L(y,z) &= \sum_l E_L^{(l)}(y,z)\exp(iK_l y)\exp(-iK_l^2 z/2k_L),\\ E_S(y,z) &= \sum_l E_S^{(l)}(y,z)\exp(iK_l y)\exp(-iK_l^2 z/2k_S). \qquad (5)\end{aligned}$$

Here, l refers to the beam number. The quantities $E_L^{(l)}$ and $E_S^{(l)}$ give the envelopes of the individual beams; these envelopes are slowly varying in the y-direction. For the case of interest to us here, it is appropriate to assume that $K_l = lK_1$, that $E_L^{(0)} = 0$, and that $E_S^{(l)}$ is very small except for $l = 0$. The $E_S^{(l)}$ for $l \neq 0$ correspond to the side beams, and it is their growth which we wish to determine.

It now follows that

$$\frac{\partial E_L^{(l)}}{\partial z} + \frac{K_l}{k_L}\frac{\partial E_L^{(l)}}{\partial y} - \frac{i}{2k_L}\frac{\partial^2 E_L^{(l)}}{\partial y^2} = 0, \qquad (6)$$

and

$$\begin{aligned}\frac{\partial E_S^{(l)}}{\partial z} + \frac{K_l}{k_S}\frac{\partial E_S^{(l)}}{\partial y} - \frac{i}{2k_S}\frac{\partial^2 E_S^{(l)}}{\partial y^2} &= \frac{g}{2}\sum_{m,n,o} E_L^{(m)} E_L^{(n)*} E_S^{(o)} \exp[i(K_m - K_n + K_o - K_l)y] \qquad (7)\\ &\quad \exp\left[\frac{-i}{2k_L}\left(K_m^2 - K_n^2 + \frac{k_L}{k_S}K_o^2 - \frac{k_L}{k_S}K_l^2\right)z\right].\end{aligned}$$

In the previous sum, we only keep terms for which

$$K_m - K_n + K_o - K_l = 0, \qquad (8)$$

or $m - n + o - l = 0$, in order to satisfy the condition that $E_S^{(l)}$ vary slowly compared with $\exp(iK_1 y)$ which in turn comes from the condition $(\Delta K)_{\rm beam} \ll (\Delta K)_{\rm sep}$.

In general, the explicit variation in Eq. (7) can lead to rapidly oscillating terms; these terms make no contribution to the sum. Writing the e-folding growth length as $(g\langle I_L\rangle)^{-1}$, where $\langle I_L\rangle$ is the summed, average strength of the pump beams in the interaction region, our condition to have rapidly oscillating terms is

$$\frac{1}{2k_L}(\Delta K)^2_{\rm sep} \gg g\langle I_L\rangle, \qquad (9)$$

a condition which is well-obeyed. A similar condition applies in the theory of Brillouin four wave mixing[1,2] or Flusberg and Korff's theory of collinear Raman interactions, although $(\Delta K)_{\rm sep}$ is replaced by the total bandwidth. In these theories, one also assumes that a complementary condition

$$\frac{r}{2k_S}(\Delta K)^2_{\rm sep} \ll g\langle I_L\rangle, \qquad (10)$$

holds, where $r = (k_L - k_S)/k_L$. As a consequence, k_L can be set equal to k_S, and we can avoid rapid oscillations when

$$K_m^2 - K_n^2 + K_o^2 - K_l^2 = 0 \qquad (11)$$

which, combining with Eq. (8), implies either 1) $K_m = K_n$ and $K_0 = K_l$ or 2) $K_m = K_l$ and $K_n = K_o$. The first case corresponds to terms in the equations which lead to intensity amplification of the Stokes wave; the second case corresponds to terms which lead to replication of the pump structure. There are as many terms of the second type as there are of the first; hence, Flusberg and Korff[11] conclude that the portion of the Stokes beam in phase with the pump grows at twice the rate of the rest of the Stokes structure.

In our experiments, Eq. (10) does not hold because of the large difference between k_L and k_S. Instead, we find

$$\frac{r}{2k_S}(\Delta K)^2_{\rm sep} > g\langle I_L\rangle \qquad (12)$$

and

$$r(\Delta K)_{\rm sep} > (\Delta K)_{\rm beam}. \qquad (13)$$

As a consequence, rapid oscillations do not occur when either 1) $K_m = K_n$ and $K_o = K_l$, just as before, or 2) $K_m = K_l = -K_n = -K_o$, which strongly restricts the previous second case. The second condition, Eq. (13), ensures that the finite bandwidth of the $E_S^{(l)}$ does not lead to a non-zero contribution from one of the terms for which $K_m = K_l$ and $K_n = K_o$, but $K_m \neq -K_n$. We now find that

Eq. (7) becomes

$$\frac{\partial E_S^{(l)}}{\partial z} + \frac{K_l}{k_S}\frac{\partial E_S^{(l)}}{\partial y} - \frac{i}{2k_S}\frac{\partial^2 E_S^{(l)}}{\partial y^2} = \frac{g}{2}\sum_m E_L^{(m)} E_L^{(m)*} E_S^{(l)} + \frac{g}{2} E_L^{(l)} E_L^{(-l)*} E_S^{(-l)}. \tag{14}$$

In the experiments of interest to us, it is always the case that

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} < g\langle I_L\rangle. \tag{15}$$

We may thus assume that over one growth length the effect of diffraction can be ignored. Letting $z_l = z$ and $y_l = y - (K_l/k_S)z$, we obtain

$$\frac{\partial E_S^{(l)}}{\partial z_l} = \frac{g}{2}\sum_m E_L^{(m)} E_L^{(m)*} E_S^{(l)} + \frac{g}{2} E_L^{(l)} E_L^{(-l)*} E_S^{(-l)}. \tag{16}$$

The quantity y_l measures transverse length from the center of the lth beam. The other beams' variation in z_l will in most cases be more rapid than that of the lth beam.

To analyze Eq. (16), we first consider a limiting case where FN_A is very long for the pump beams, and, at a given y_l, their amplitude variation as a function of z_l can be neglected over some long length in the interaction region. Equation (16) then has the solution for $l = 0$,

$$E_S^{(0)}(z_l, y_l) = E_S^{(0)}(0, y_l)\exp\left[\frac{g}{2}\langle I_L\rangle\right], \tag{17}$$

where $\langle I_L\rangle = \sum_m E_L^{(m)} E_L^{(m)*}$, and we recall $E_L^{(0)} = 0$. When $l \neq 0$, the equations for l and $-l$ are coupled, and, assuming that

$$|E_L^{(l)*} E_L^{(-l)}| = \langle I_L\rangle/N,$$

where N is the number of beams, we find

$$\begin{pmatrix} E_S^{(l)} \\ E_S^{(-l)} \end{pmatrix} = \alpha\begin{pmatrix} \exp(i\phi_l) \\ 1 \end{pmatrix}\exp\left[\frac{g}{2}\frac{N+1}{N}\langle I_L\rangle z_l\right] + \beta\begin{pmatrix} \exp(i\phi_l) \\ -1 \end{pmatrix}\exp\left[\frac{g}{2}\frac{N-1}{N}\langle I_L\rangle z_l\right], \tag{18}$$

where $\exp(i\phi_l) = E_L^{(l)} E_L^{(-l)*}/|E_L^{(l)} E_L^{(-l)*}|$, and

$$\alpha = \frac{1}{2}\left[E_S^{(l)}(0, y_l)\exp(-i\phi_l) + E_S^{(-l)}(0, y_l)\right],$$
$$\beta = \frac{1}{2}\left[E_S^{(l)}(0, y_l)\exp(-i\phi_l) - E_S^{(-l)}(0, y_l)\right]. \tag{19}$$

We find that the vector $(E_S^{(l)}, E_S^{(-l)})^t$ consists of two portions, a portion which is in phase with $(E_L^{(l)}, E_L^{(-l)})^t$ and grows somewhat faster than the central Stokes beam and a portion which is out of phase with $(E_L^{(l)}, E_L^{(-l)})^t$ and grows somewhat slower. On a length scale longer than $(g\langle I_L\rangle/N)^{-1}$, the in phase component dominates over the out of phase component.

At this point, we can outline the condition for side beam replication to occur. If the pump beams satisfy the condtion

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} < g\langle I_L\rangle/N, \tag{20}$$

then the pump beams $E_L^{(l)}$ and $E_L^{(-l)}$ are correlated over a length greater than $(g\langle I_L\rangle/N)^{-1}$ and, as a result, the phase difference between $E_S^{(l)}$ and $E_S^{(-l)}$ is locked to the phase difference between the pump beams. Thus, the gain of the side beams is higher than that of the central Stokes beam, and side beams will be observable if the overall gain is sufficiently large. By contrast, in the opposite limit of Eq. (20), the phases of $E_L^{(l)}$ and $E_L^{(-l)}$ change too rapidly for the phase difference of the Stokes beams to lock to them. In this case, the average growth rate of the side beams is only slightly higher then that of the central beam, and side beam replication is not expected to occur.

3. DISCUSSION

The experiments of interest to us here were carried out at the Naval Research Laboratory using a XeCl laser at 308 nm and a high pressure H_2 cell.[9,10] The Stokes radiation emerges at 353 nm implying that $r = (k_L - k_S)/k_L = 0.13$. The angular separation between incoming pump beams is 5.6 mrad, so that $(\Delta K)_{\text{sep}} = 1.1\times10^3$ cm^{-1}. Experiments were carried out with pump beams 20 or 120 times dispersion limited. In the former case, their angular spread was typically 0.03 mrad and in the latter case, it was typically 0.18 mrad, corresponding respectively to $(\Delta K)_{\text{beam}} = 6$ cm^{-1} and $(\Delta K)_{\text{beam}} = 37$ cm^{-1}. The interaction length of the H_2 chamber is 500 cm and in all cases $4 < g\langle I_L\rangle z < 20$, so that there is enough gain to achieve reasonable amplification without causing self-oscillation. We conclude $8\times10^{-3} < g\langle I_L\rangle < 4\times10^{-2}$. The number of beams is given by $N = 24$.

We now examine our conditions to be sure that they are met. We first find

$$\frac{r}{2k_S}(\Delta K)^2_{\text{sep}} = 0.5\ \text{cm}^{-1} > g\langle I_L\rangle. \tag{12'}$$

$$r(\Delta K)_{\text{sep}} = 150\ \text{cm}^{-1} > (\Delta K)_{\text{beam}} = 6 - 37\ \text{cm}^{-1}. \tag{13'}$$

We next examine $(1/2k_L)(\Delta K)^2_{\text{beam}}$. For the 20 times dispersion limited beam we find

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} = 9\times10^{-5}\ \text{cm}^{-1} < g\langle I_L\rangle, \tag{15'}$$

and for the 120 times dispersion limited beam, we find

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} = 3.4\times10^{-3}\ \text{cm}^{-1} < g\langle I_L\rangle. \tag{15''}$$

Thus, all our basic conditions are met. We now recall $N = 24$, so that

$$g\langle I_L\rangle/N = 3\times10^{-4} - 1.7\times10^{-3}.$$

When the pump beams are 20 times dispersion limited, we thus find

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} < g\langle I_L\rangle/N, \tag{20'}$$

and we expect side beam replication to occur. By contrast, when the pump beams are 120 times dispersion limited, we find

$$\frac{1}{2k_L}(\Delta K)^2_{\text{beam}} > g\langle I_L\rangle/N, \tag{20''}$$

and no pump replication is expected. Both these results are in accord with the experiments.

We note that while the theory of Sec. 2 agrees well with the experiments, it is not sufficiently refined to lead to a precise determination of the boundaries between the different regimes. Here, numerical simulations are likely to be of assistance, and we intend to carry them out in the near future.

Finally, we turn to methods for eliminating side beam replication. These include: 1) Since $E_L^{(1)}$ and $E_L^{(-1)}$ must both be non-zero for pump beam replication to occur, we can arrange the pump beams asymmetrically. Unfortunately, this approach lead to asymmetries in the Stokes amplification and degradation of the beam quality. 2) We can increase the number of pump beams. This approach does not appear to be practical. 3) We can phase $E_L^{(1)}$ and $E_L^{(-1)}$ so that they are out of phase with each other. If, as seems likely, the Stokes beams are seeded almost symmetrically by scattering from the central beam, the pump and Stokes beams should be out of phase. This approach appears promising, and we intend to explore it.

4. ACKNOWLEDGMENT

The work of Drs. Menyuk and Hilfer was supported by the Naval Research Laboratory.

* Permanent address: Department of Electrical Engineering, University of Maryland, Baltimore, MD 21228.

5. REFERENCES

1. R. W. Hellwarth, "Theory of phase conjugation by stimulated scattering in a waveguide," J. Opt. Soc. Am. **68**(8), 1050–1056 (1978).
2. B. Ya. Zel'dovich, N. F. Pilipetskiĭ, and V. V. Shkunov, "Phase conjugation in stimulated scattering," Usp. Fiz. Nauk **138**(10), 249–288 (1982) [Sov. Phys. Usp. **25**(10), 713–737 (1982)].
3. J. Goldhar and J. R. Murray, "Intensity averaging and four-wave mixing in Raman amplifiers," J. Quantum Electron. **QE-18**(3), 399–409 (1982).
4. R. S. F. Chang and N. Djeu, "Amplification of a diffraction-limited Stokes beam by a severely distorted pump," Opt. Lett. **8**(3), 139–141 (1983).
5. V. I. Bespalov, A. A. Betin, and G. A. Pismanik, "Reproduction of the pumping wave in stimulated-scattering radiation," Radiophys. Quantum Electron. **21**(7), 675–688 (1978) [Izv. Vyssh. Uchen. Zaved. Radiofiz. **21**(7), 961–980 (1978)].
6. J. Goldhar, M. W. Taylor, and J. R. Murray, "An efficient double-pass Raman amplifier with pump intensity averaging in a light guide," IEEE J. Quantum Electron. **QE-20**(7), 772–785 (1984).
7. R. S. F. Chang, M. T. Duignan, R. H. Lehmberg, and N. Djeu, "Use of stimulated Raman scattering for reducing the divergence of severely aberrated laser beams," in *Excimer Lasers, Their Applications, and New Frontiers in Lasers*, Proc. SPIE **476**, 81–89 (1984).
8. R. S. F. Chang, R. H. Lehmberg, M. T. Duignan, and N. Djeu, "Raman beam cleanup of a severely aberrated pump laser," J. Quantum Electron. **QE-21**(5), 477–487 (1985).
9. J. Reintjes, R. H. Lehmberg, G. Calame, and M. T. Duignan, "Applications of Raman beam cleanup and phase conjugation to the reduction of laser beam aberrations," in *Laser Research and Development in the Northeast*, Proc. SPIE **709**, 20–27 (1986).
10. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam cleanup with stimulated Raman scattering in the intensity-averaging regime," J. Opt. Soc. Am. B **3**(10), 1408–1427 (1986).
11. A. Flusberg and D. Korff, "Wave-front replication versus beam cleanup by stimulated scattering," J. Opt. Soc. Am. B **4**(5), 687–690 (1987).

"Asymptotic evolution of transient pulses undergoing stimulated Raman scattering," (C.R. Menyuk and G. Hilfer), to be submitted to Optics Lett.

ASYMPTOTIC EVOLUTION OF TRANSIENT PULSES UNDERGOING STIMULATED RAMAN SCATTERING

by

Curtis R. Menyuk* and Godehard Hilfer
Science Applications International Corporation
1710 Goodridge Drive
McLean, VA 22102

Abstract

Propagation of short, transient pulses undergoing stimulated Raman scattering over long length scales is considered. It is shown that under common experimental circumstances, the evolution has two different regimes: 1) The I-regime, at short lengths, where the pump changes little and the Stokes rapidly grows, and 2) the J-regime, at long lengths, where the Stokes intensity is close to saturation and the pump intensity decreases slowly as the square root of distance. The distance at which the J-regime is reached is determined numerically.

*Permanent Address: Department of Electrical Engineering
University of Maryland
Catonsville, MD 21228

Asymptotic Evolution of Transient Pulses Undergoing Stimulated Raman Scattering

Since the early work of Carmen, *et al.*[1] the evolution of pulses undergoing stimulated Raman scattering has been a subject of constant interest.[2-12] In the limit considered by Carmen, *et al.*[2] where diffraction, level saturation, interaction with anti-Stokes or higher order Stokes radiation, and quantum noise can all be ignored, the wave interaction is governed by the equations

$$\begin{aligned} \frac{\partial E_L}{\partial z} &= -i\frac{k_L}{k_S}\kappa_2 Q E_S \quad , \\ \frac{\partial E_S}{\partial z} &= -i\kappa_2 Q^* E_L \quad , \\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i\kappa_1 E_S^* E_L \quad . \end{aligned} \tag{1}$$

The purpose of this letter is to revisit Eq. (1) in the highly transient limit where $T_2/\tau \ll 1$. The quantity $T_2 = \Gamma^{-1}$ is the damping time of the material excitation, and τ is the full width at half maximum (FWHM) pulse intensity. This limit is relevant to recent experiments which have been carried out at the Naval Research Laboratory.[11,12]

It has long been known that in the initial growth phase where $|E_S| \ll |E_L|$, Eq. (1) can be linearized in a simple way, allowing for a simple characterization of the solution. We shall show that in the limit of large z, a simple description is once again possible. In effect,

$$K(t) \equiv |E_L(z,t)|^2 + \frac{k_L}{k_S}|E_S(z,t)|^2 \tag{2}$$

remains constant for all z. We thus find that as the pump intensity diminishes, the Stokes intensity grows, ultimately taking on the shape of the initial pump. One can then assume that the Stokes intensity is fixed and carry out a theory analogous to that of Carmen, *et al.*[1] One finds, however, that the I-Bessel functions are replaced by J-Bessel functions.

Recent experiments at the Naval Research Laboratory have studied transient pulses short compared to T_2.[11,12]. These pulses typically have a slight chirp proportional to the pump intensity, but no rapid phase shift. Under these circumstances, numerical results indicate that there is a rapid transition between the I-regime where the theory of Carmen, *et al.*[1] applies and the J-regime where the theory to be presented shortly applies. In other experimental settings, where a phase shift which is rapid compared to the pulse size is present, a soliton-like structure can form;[7,10] however, its velocity is smaller than light, so that it must ultimately travel to the back end of the pulse and disappear if the pulse size is short compared to T_2. Moreover, we will show that soliton-like structures cannot form when the pulse size is short compared to T_2 if two conditions are met: 1) the initial Stokes amplitude is small compared to the pump, and 2) there is no phase variation in the leading edge of the pulse.

We begin our theoretical development by recalling that in the I-regime, the solution to Eq. (1) is given by

$$\begin{aligned} E_S(z,t) = E_S(0,t) + (\kappa_1\kappa_2 z)^{1/2} E_L(t) \int_{-\infty}^{t} \exp[-\Gamma(t-t')] \\ E_L^*(t') E_S(0,t') [\tau(t) - \tau(t')]^{-1/2} I_1\left(2\{\kappa_1\kappa_2 z[\tau(t) - \tau(t')]\}^{1/2}\right) dt' \quad , \end{aligned} \tag{3a}$$

$$\begin{aligned} Q(z,t) = -i\kappa_1 \int_{-\infty}^{t} \exp[-\Gamma(t-t')] E_L(t') E_S^*(0,t') \\ I_0\left(2\{\kappa_1\kappa_2 z[\tau(t) - \tau(t')]\}^{1/2}\right) dt' \quad , \end{aligned} \tag{3b}$$

where

$$\tau(t) = \int_{-\infty}^{t} K(t')\, dt' \quad . \tag{4}$$

We now solve Eq. (3a) approximately using the method of steepest descent.[13] In the regime which we are considering, where T_2 is much larger than the pulse width, most of the contribution to the integral comes from a restricted region in t' where the rapid increase in E_L and/or E_S at their leading edges balances the rapid decrease in the Bessel function as $\tau(t')$ approaches $\tau(t)$. The steepest descent path is along the real t-axis. The details of the solution depend on the rapidity with which E_L and E_S vary in the neighborhood of the steepest descent point.

We now assume that the initial Stokes pulse leads the pump pulse and is varying slowly at the steepest descent point; this assumption corresponds to maximum gain.[11,12] We will further assume that leading edge of the pump varies exponentially. We define now

$$s \equiv 4\kappa_1\kappa_2 z[\tau(t) - \tau(t')] \ ,$$
$$s_\infty \equiv 4\kappa_1\kappa_2 z\tau(t) \ , \tag{5}$$

and note that when s is large

$$I_1(s^{1/2}) \simeq \exp[s^{1/2} - \frac{1}{2}\ln(2\pi s^{1/2})] \ . \tag{6}$$

Physically, we are assuming that z is large enough so that the Stokes pulse has undergone substantial gain, but is not so large that pump depletion has begun. For these assumptions to be consistent, the initial Stokes amplitude must be small relative to the pump amplitude. At the leading edge of the pump pulse, we write by assumption

$$E_L(t') = A_L \exp(\Gamma_w t') \exp[i\phi_L(t')] \ , \tag{7}$$

where A_L, Γ_w and ϕ_L are all real. Equation (7) effectively defines all three quantities. It is useful to define another quantity $r(t)$ through the relationship

$$\tau(t) = r^2(t) A_L^2/\Gamma_w \ . \tag{8}$$

We stress that s, and thus the steepest descent point t_0, is a function of t.

In the case being considered, we may write $E_S(t') = A_S \exp[i\phi_S(t')]$. Both phases ϕ_L and ϕ_S are assumed to be slowly varying. Gathering together all the rapidly varying terms and substituting the results into Eq. (3a), we find that the argument of the resulting exponent is given by

$$\psi = (\Gamma_w + \Gamma)t' + s^{1/2} - \frac{1}{2}\ln(2\pi s^{3/2}) \ . \tag{9}$$

The steepest descent point is the point at which $d\psi/dt' = 0$. This point satisfies the relation

$$\Gamma_w + \Gamma + \frac{3}{4}\ \frac{|E_L|^2(t')}{[\tau(t) - \tau(t')]} - \frac{(\kappa_1\kappa_2 z)^{1/2}|E_L|^2(t')}{[\tau(t) - \tau(t')]^{1/2}} = 0 \ . \tag{10}$$

At large z with t inside the main part of the pulse, t' is out on the leading edge of the pulse; hence, $\tau(t') \ll \tau(t)$ and may be neglected to lowest order in $s_\infty^{1/2}$. Using Eq. (7), we conclude

$$t_0 = \frac{1}{2\Gamma_w} \ln\left[\frac{2(1+\Gamma/\Gamma_w)\tau^2(t)}{s_\infty^{1/2} - 3/2}\right] \ . \tag{11}$$

Carrying out the remainder of the steepest descent calculation,[13] we find

$$\begin{aligned} E_S(z,t) = 2\kappa_1\kappa_2 z \left[\frac{\pi}{\Gamma_w^2(1+\Gamma/\Gamma_w)}\right]^{1/2} E_L(t)E_S(0,t_0)E_L^*(t_0) \\ \exp[-\Gamma(t-t_0)]\exp(s^{1/2})/(2\pi s^{3/2})^{1/2} \ , \end{aligned} \tag{12}$$

where s is evaluated at $t = t_0$. We stress that this calculation is not asymptotic in z as the exponential rise of E_L is controlled by Γ_w, not z. It does, however, yield a useful approximation. We have compared Eq. (12) to numerically calculated exact solutions of Eq. (1) in several instances, and we have shown that they agree well to within factors of order unity in the appropriate parameter regime.

We can obtain a number of results directly from these calculations. First, the phase difference $\phi_L(t) - \phi_S(t)$ in the bulk of the Stokes pulse at large z is controlled by the phase difference $\phi_L[t_0(t)] - \phi_S[t_0(t)]$ at $z = 0$. At large z, the range of t_0-values controlling the bulk phases reaches a constant value. Considering the half-widths at half maxima, we find

$$\Delta t_0 = \frac{1}{2\Gamma_w} \ln[\tau^2(\tau/2)/\tau^2(-\tau/2)] \ . \tag{13}$$

Second, since the central t_0-value deceases with increasing z, we conclude that if the initial phase difference approaches a constant value, soliton-like structures cannot form. Third, for any given z and t, it follows from Eq. (12) that the maximum growth is obtained by placing

$E_S(t_0)$ at the steepest descent point. It is not trivial to determine precisely the optimum offset for the Stokes pulse from this calculation as we must sum the contribution at all values of t; however, we immediately conclude that the Stokes pulse should precede the pump pulse by an amount on the order of $1/\Gamma_w$. These conclusions all agree well with available experimental and computational results.[11,12]

We turn now to the J-regime. In this regime, where the Stokes intensity is close to its asymptotic value, we find

$$\begin{aligned} E_L(z,t) = E_L(z_0,t) - [\kappa_1\kappa_2(z-z_0)]^{1/2}\frac{k_L}{k_S}E_S(t) \\ \int_{-\infty}^{t} \exp[-\Gamma(t-t')]E_S^*(t')E_L(0,t') \\ [\tau(t)-\tau(t')]^{-1/2}J_1\Big(2\{\kappa_1\kappa_2(z-z_0)[\tau(t)-\tau(t')]\}^{1/2}\Big)\,dt' \ , \end{aligned} \tag{14a}$$

$$\begin{aligned} Q(z,t) = -i\kappa_1\int_{-\infty}^{t}\exp[-\Gamma(t-t')]E_S^*(t')E_L(z_0,t') \\ J_0\Big(2\{\kappa_1\kappa_2(z-z_0)[\tau(t)-\tau(t')]\}^{1/2}\Big)\,dt' \ , \end{aligned} \tag{14b}$$

These equations can be derived using the approach described by Wang[14] or verified by substitution into Eq. (1). Using the asymptotic expression

$$J_n(x) = \left(\frac{2}{\pi x}\right)^{1/2}\cos(x - \frac{1}{2}n\pi - \frac{1}{4}\pi) \ , \tag{15}$$

we reach the following conclusions: First, at large z, the amplitude E_L at any time scales like $z^{-1/4}$, multiplied by a periodic variation. The total integrated intensity must therefore scale as $z^{-1/2}$. Second, the number of zero-crossings of the real and imaginary parts of E_L and Q scale like $z^{1/2}$. Third, new zeros enter the E_L and Q pulses at large t and travel toward smaller t as z increases.

In order to verify these trends, we have considered a large number of numerical solutions which will be presented in detail in a later publication. A small fraction of these results are summarized in Figs. 1 and 2. In these figures we display $R = [\int_{-\infty}^{\infty} dt\, |E_L|^2(0) / \int_{-\infty}^{\infty} dt\, |E_L|^2(\varsigma)]^2$ vs. $\varsigma \equiv \kappa_1 \kappa_2 z r(\infty)$ and N (the number of zero-crossings of E_L) vs. ς, where N is plotted on a parabolic scale. We display three different cases, in all of which E_L and E_S are purely real and Q is purely imaginary. The initial Stokes intensity is 10^{-3} of the pump intensity at all points in time. The maximum intensity of the pump is the same in all three examples, and their FWHM values of the the initial pump profiles are chosen so that they all have nearly the same integrated intensity. In all cases, the FWHM is roughly 40 ps and $T_2 = 633$ ps. We find that the observed scaling agrees well with the analytical predictions. Moreover, for all the cases shown here, the linear scaling is obtained when $\varsigma \simeq 120$ and $R \simeq 10$, corresponding to 70% pump depletion. In our examination of a large number of different cases, we have noted the following trends for pulses short compared to T_2: 1) There is little dependence on the pulse shape. 2) When the Stokes offset is decreased, *i.e.*, the Stokes pulse arrives at the Raman cell earlier than the pump pulse, the pump must deplete more before the J-regime is reached. With a negative offset equal to the FWHM, the pump must be 90% depleted before R scales linearly. The scaling of N is, however, only slightly affected. Moreover, the ς-value at which the J-regime is reached only increases from $\varsigma = 120$ to $\varsigma = 180$. 3) When the Stokes offset is increased so that the Stokes pulses arrives after the pump, one finds that beyond 70% pump depletion R scales linearly. However, linear scaling of N^2 is delayed. With a positive offset equal to the FWHM, this scaling sets in at around $\varsigma = 180$. 4) When a chirp proportional to the pump amplitude is added to the pump and/or Stokes, no effect is observed when the magnitude of the chirp is approximately π, the experimental value. When the magnitude increases to approximately 10π, the ς-value at which the J-regime is reached increases slightly, by under 50, at all Stokes offsets, with no observed alteration in the basic trends.

In this letter, we have considered the effect of stimulated Raman scattering on short, transient pulses. We have shown that under normal experimental circumstances where the phase difference between the pump and Stokes pulses at their leading edges reaches a constant and where the Stokes pulse precedes the pump, the Stokes pulse will phase lock to the pump and grow steadily. Once the Stokes has nearly saturated, the total pulse enters a new regime where the pump intensity decreases as the square root of distance and the pump amplitude oscillates with a frequency proportional to the square of the distance.

This work was supported by the Naval Research Laboratory, and we gratefully acknowledge useful conversations with Dr. J. Reintjes.

REFERENCES

1. R. L. Carmen, F. Shimizu, C. S. Wang, and N. Bloembergen, Phys. Rev. A **2**, 60 (1970).

2. N. Tan-no, T. Shirahata, K. Yokoto, and H. Inaba, Phys. Rev. A **12**, 159 (1975).

3. G. I. Kachen and W. H. Lowdermilk, Phys. Rev. A **14**, 1472 (1976).

4. J. N. Elgin and T. B. O'Hare, J. Phys. B **12**, 159 (1979).

5. M. G. Raymer, J. Mostowski, and J. L. Carlsten, Phys. Rev. A **19**, 2304 (1979).

6. M. G. Raymer and J. Mostowski, Phys. Rev. A **24**, 1980 (1981).

7. F. Y. F. Chu and A. C. Scott, Phys. Rev. A **12**, 2060 (1975).

8. K. Druhl, R. G. Wenzel, and J. L. Carlsten, Phys. Rev. Lett. **51**, 1171 (1983).

9. D. J. Kaup, Physica **6D**, 143 (1983).

10. H. Steudel, Physica **6D**, 155 (1983).

11. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, JOSA B **5**, 37 (1988).

12. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes in *Beam Conbining and Atmospheric Propagation* (Proc. SPIE 784, Bellingham, WA, to appear).

13. See, *e.g.* P. M. Morse and H. Feshbach, *Methods of Theoretical Physics* (McGraw, New York, 1953), pp. 434–443.

14. C. S. Wang, Phys. Rev. **182**, 482 (1969).

FIGURE CAPTIONS

1. Plots of R vs. ς for different pulse shapes. a) sech-squared amplitude, FWHM = 40 ps; b) Lorentzian-squared amplitude, FWHM = 39 ps; c) Square pulse, FWHM = 43.8 ps.
2. Plots of N vs. ς; N is plotted on a parabolic axis. Shapes and parameters are the same as in Fig. 1.

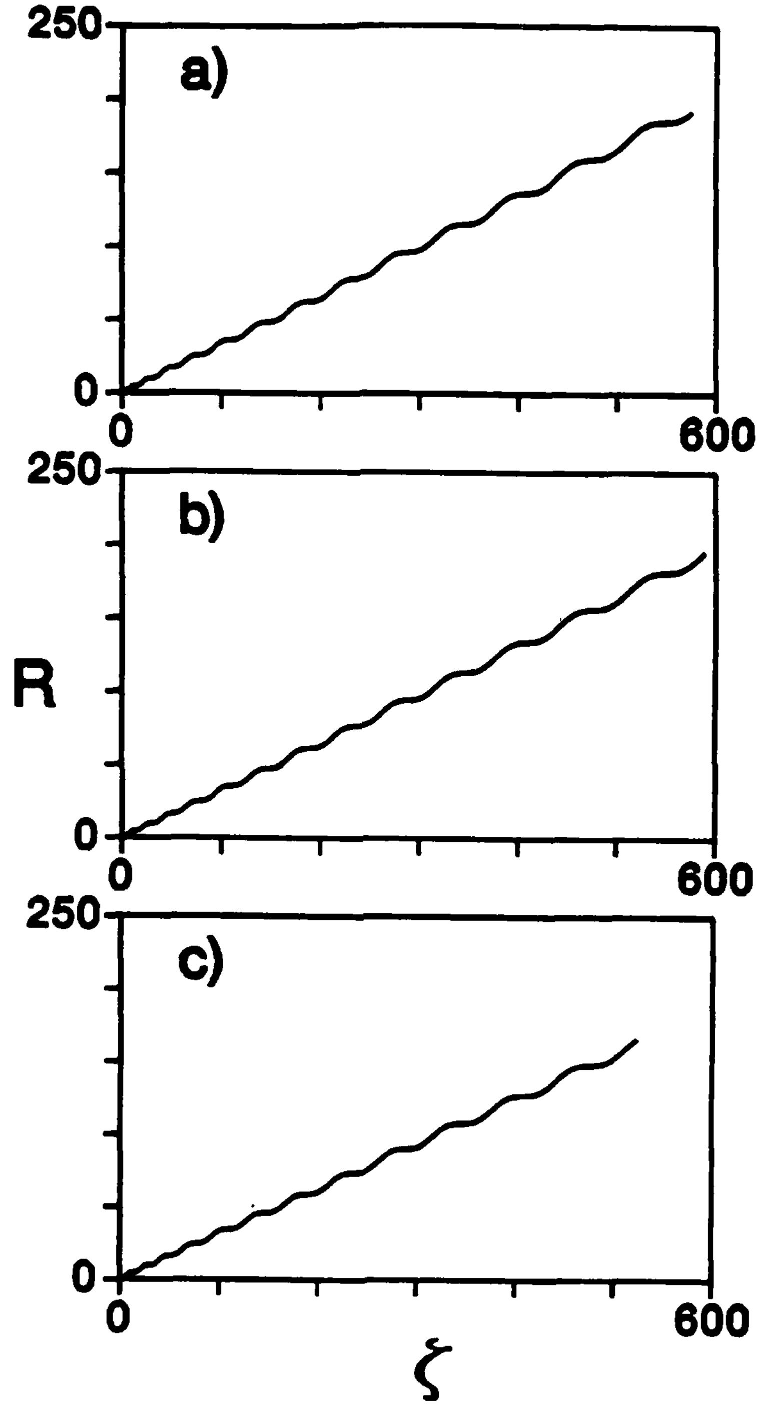


FIGURE 1.

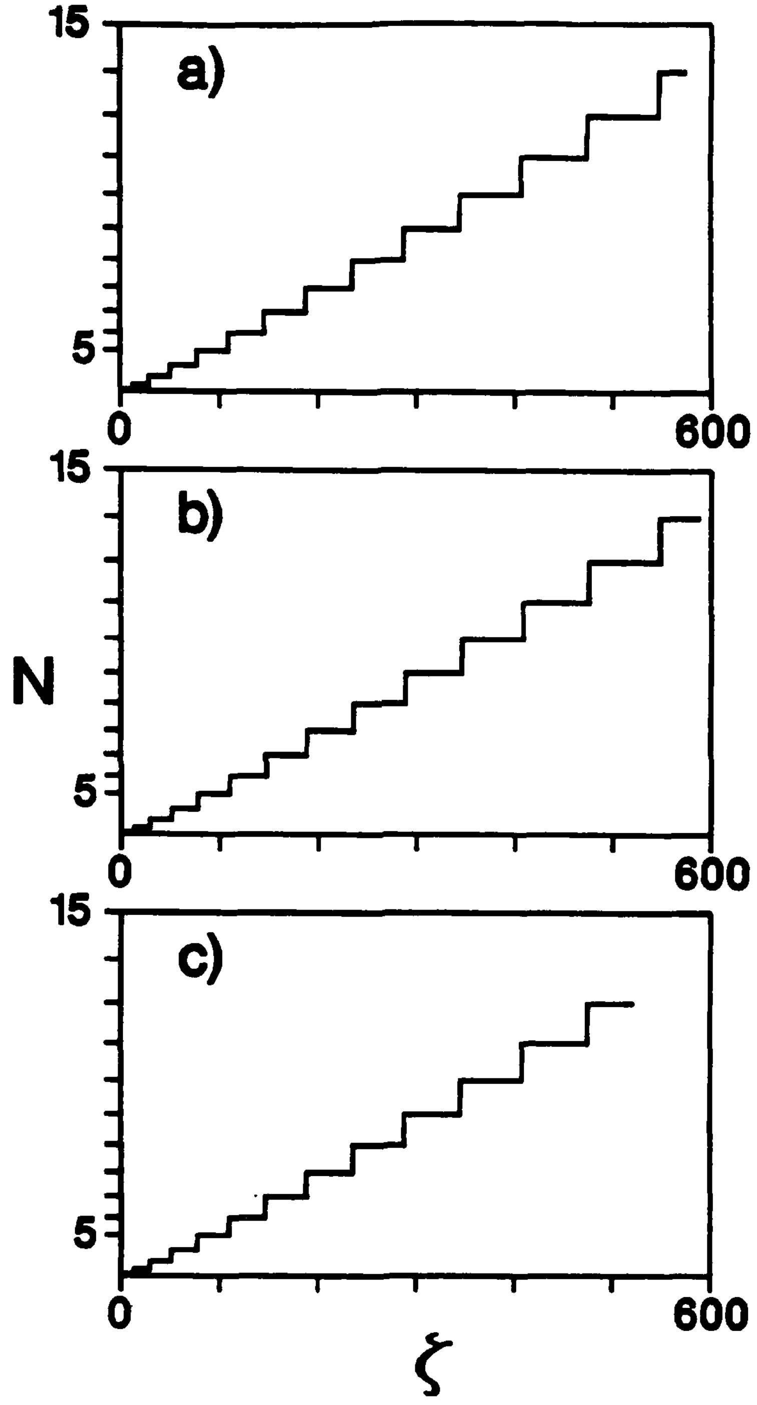


FIGURE 2.

APPENDIX D

Presentations

IVth Workshop on Nonlinear Evolution Equations and Dynamical Systems, (Balaruc-les-Bains, France, June 6-25, 1978).

LIE PERTURBATION METHODS AND THEIR APPLICATION TO INFINITE-DIMENSIONAL, HAMILTONIAN SYSTEMS

Curtis R. MENYUK

University of Maryland -
Department of Electrical
Engineering -
COLLEGE PARK, MD 20742
U.S.A.

ABSTRACT

The Lie perturbation method of Hori and Deprit is a practical approach for determining the evolution of finite-dimensional, nearly integrable, Hamiltonian systems. It has been applied with notable success to problems including satellite motion around the Earth and particle motion in accelerators. We review this approach and describe how to extend it to infinite-dimensional systems. Explicit first and second order calculations are described in cases where the initial data contains a single solitary wave and a small amount of radiation. Implications for the emergence of solitary waves from arbitrary initial data are discussed.

REFERENCES

1. C.R. Menyuk and H.H. CHEN, On the Hamiltonian structure of ion acoustic waves and shallow channel water waves, Phys. Fluids 29, 998-1003 (1986).
2. C.R. MENYUK, Origin of solitons in the "real" world, Phys. Rev. A33, 4367-4374 (1986).
3. C.R. MENYUK, Nonlinear pulse propagation in optical fibers, I.E.E.E. J. Quantum Electron. QE-23, 174-176 (1986).

Anaual Meeting of the Optical Society of America (Rochester, NY, October 18-23, 1987), papers MI7 and MI12.

MI7 Numerical modeling of transient Raman amplification

GODEHARD HILFER, Science Applications International Corp., 1710 Goodridge Dr., McLean, VA 22102; CURTIS R. MENYUK, U Maryland, College Park, MD 20742.

To model experiments performed by Reintjes *et al.*,[1] a numerical code has been developed that solves the Raman interaction equations:

$$\frac{\partial A_L}{\partial z} - \frac{i}{2k_L}\frac{\partial^2 A_L}{\partial y^2} = i\kappa_2 \frac{\omega_L}{\omega_S} A_S Q,$$

$$\frac{\partial A_S}{\partial z} - \frac{i}{2k_S}\frac{\partial^2 A_S}{\partial y^2} = i\kappa_2 A_L Q^*,$$

$$\frac{\partial Q}{\partial t} + \Gamma Q = i\kappa_1 A_L^* A_S.$$

The purpose is to study the influence of transience, transverse beam structure, pump depletion, and dispersive effects on the amplification and phase modulation of diverse forms of transient Stokes and pump beams in a cross-beam geometry.

Preliminary results of these simulations are reported for the beam parameters of the NRL experiments **(12 min)**

1. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam Cleanup with Stimulated Raman Scattering in the Intensity-Averaging Regime," J. Opt. Soc. Am. B 3, 1408 (1986), see Table 1.

MI12 Linear theory of Raman beam cleanup and amplification in a crossing beam geometry

CURTIS R. MENYUK, U Maryland, Department of Electrical Engineering, Catonsville, MD 21228

In experiments which have been performed to date at the Naval Research Laboratory,[1] it has been discovered that the results can be characterized by the Fresnel number of the aberrations $D_A^2/\lambda L$, where D_A is the transverse scale length of the aberrations, L is the interaction length of the pump and Stokes beams, and λ is the pump wavelength. The results are characterized by the geometry (colinear or crossing beam) and aberrations (phase, amplitude, or both). We show that many of the experimentally observed effects can be explained by a linear theory, although nonlinear effects due to pump depletion are in most cases important. The circumstances in which off-angle contributions, copropagating with the crossing beams, are seeded by the pump and can grow to important levels are elucidated. **(12 min)**

1. J. Reintjes, R. H. Lehmberg, R. S. F. Chang, M. T. Duignan, and G. Calame, "Beam Cleanup with Stimulated Raman Scattering in the Intensity-Averaging Regime," J. Opt. Soc. Am. B 3, 1408 (1986), see Table 1.

Workshop on Solitons and Chaos in Optical Systems (San Jose, CA, January 6-7, 1988).

SOLITONS IN A KERR MEDIUM

Curtis R. Menyuk
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228

ABSTRACT: In a weakly dispersive medium with a Kerr nonlinearity, the equations governing the electromagnetic wave evolution can be written as

$$u_\xi + \frac{1}{2}u_{ss} + (|u|^2 + B|v|^2)u = 0,$$
$$v_\xi + \frac{1}{2}v_{ss} + (B|u|^2 + |v|^2)v = 0,$$

where u and v are the envelopes of the two polarization amplitudes, ξ is the normalized distance, s is the normalized time, and B depends on the medium in question. For media such as optical fibers, where the Kerr response is essentially instantaneous, $B = 2/3$ [See, *e.g.* C. R. Menyuk, IEEE J. Quantum. Electron. **QE-23**, 174 (1986)]. Painlevé analysis indicates that this system is only integrable when $B = 0$ or $B = 1$. Both these cases merit profound study since they provide a useful starting point for studying the general case where $B \neq 0$ or 1. In the former case, this system reduces to two uncoupled nonlinear Schrödinger equations. The nonlinear Schrödinger equation has been extensively studied. In the latter case, this system has been studied by Manakov [Sov. Phys. JETP **38**, 248 (1974)] who found the Lax pair for this system, showed how to solve the Cauchy problem using spectral transform methods, and explicitly derived single soliton solutions where both u and v are proportional to sech(αs). In this work it is shown by a direct search for stationary solutions that the class of single soliton solutions is substantially larger than the sech-like class found by Manakov. The soliton profiles can be obtained in the general case by using an approach originally described by Darboux [Archives néerlandaises, Ser. 2, 6, 371 (1901)].

SPIE: O-E LASE '88. Nonlinear Optics and Beam Combining (Los Angeles, CA, January 11-15, 1988), paper 874-50.

TUESDAY 12 JANUARY 1988

Registration and Information, Hilton Pavilion . . . . 7:15 am to 6:00 pm
Central Message Desk, Hilton Pavilion . . . . . 7:15 am to 6:00 pm
Information Desk, Marriott Ballroom Lobby . . . . 7:15 am to 6:00 pm
Breakfast Breads and Coffee, Marriott Ballroom Lobby . . . . 7:30 to 8:30 am
Speakers' Audiovisual Desk, Marriott Ballroom Lobby . . . . 7:30 am to 5:00 pm
Placement Service Center, Hilton International Ballroom . . . . 10:00 am to 4:00 pm
Exhibits, Hilton Pavilion Hilton International Ballroom . . . . 10:00 am to 6:00 pm

SESSION 3 . . . . Tues. 8:15 am

Nonlinear Optics and Beam Combining III

Chair: **Matthew B. White,** Office of Naval Research

Invited Paper: **Raman beam combining using broadband XeCl laser radiation,** M. N. Ediger, J. F. Reintjes, Naval Research Lab. . . . . [874-13]

Nonlinear hydrodynamic effects in gaseous SBS media, D. M. Walsh, B. S. Masson, U.S. Air Force Weapons Lab. . . . . . . [874-14]

Coherent beam processing concepts, P. Yeh, A. E. T. Chiou, I. C. McMichael, M. Khoshnevisan, Rockwell International Science Ctr. . . . . [874-15]

Characterization of asymmetric self-defocusing and centrosymmetric scattering in barium titanate, T. R. Moore, Lawrence Livermore National Lab.; D. L. Walters, U.S. Naval Post Graduate School . . . . [874-48]

***Coffee Break . . . . 10:00 to 10:30 am**

Invited Paper **Beam combining in a gas via nonlinear, diffractive optics,** J. S. Chivian, LTV Missiles and Electronic Group; C. D. Cantrell, Univ. of Texas at Dallas; W. D. Cotten, LTV Missiles and Electronics Group; C. A. Glosson, Univ. of Texas, Dallas . . . . [874-16]

High frequency stimulated Brillouin scattering experiments, M. E. Farey, C. G. Koop, TRW, Inc. . . . . [874-17]

Invited Paper **Stokes-anti-Stokes gain suppression in the transient regime,** A. B. Hickman, W. K. Bischel, SRI International . . . . [874-18]

SPIE-Hosted Picnic-style Lunch, Hilton Plaza (Lower Level) . . . . Noon to 1:00 pm
Dessert in the Exhibit Halls, Hilton Pavilion and International Ballroom . . . . 1:00 to 2:00 pm

SESSION 4 . . . . Tues. 2:00 pm

Nonlinear Optics and Beam Combining IV

Chair **Pochi Yeh,** Rockwell International Science Center

Invited Paper **Stimulated Brillouin scattering aberration control,** M. J. Lefebvre, S. J. Pfeifer, TRW, Inc . . . . [874-19]

Coherent beam combination via microparticle plasma modes, D. N. Rogovin, T. P. Shen, Rockwell International Science Ctr . . . . [874-20]

Pump replication in stimulated Raman scattering using a crossed beam geometry, C. R. Menyk, G. Hilfer, Science Applications International Corp., J. Reintjes, Naval Research Lab. . . . . [874-50]

***Coffee Break . . . . 3:40 to 4:00 pm**

Invited Paper: **Laser beam combining through the nonlinear response of a strongly driven atomic transition,** K. R. MacDonald, M. T. Gruneisen, R. W. Boyd, Univ. of Rochester . . . . [874-22]

Orientational Kerr effect for millimeter wave applications, R. L. McGraw, Rockwell International Science Ctr. . . . . [874-23]

One-way transmission of images through a multimode optical fiber by degenerate four-wave mixing in a photorefractive BSO crystal, E.-S. Kim, California Institute of Technology . . . . . [874-24]

Frequency adding media for short wavelength gases and phase-insensitive beam combinations, J. A. Goldstone, J. P. Stone, Rockwell International Corp./Rocketdyne Div. . . . . [874-25]

WEDNESDAY 13 JANUARY 1988

Breakfast Breads and Coffee Marriot Ballroom Lobby . . . . 7:30 to 8:30 am
Registration and Information, Hilton Pavilion . . . . 7:30 am to 5:00 pm
Central Message Desk, Hilton Pavilion . . . . . 7:30 am to 5:00 pm
Information Desk, Marriott Ballroom Lobby . . . . 7:30 am to 5:00 pm
Speakers' Audiovisual Desk, Marriott Ballroom Lobby . . . . 7:30 am to 5:00 pm
Placement Service Center, Hilton International Ballroom . . . . 10:00 am to 4:00 pm
Exhibits, Hilton Pavilion, Hilton International Ballroom . . . . 10:00 am to 5:00 pm

SESSION 5 . . . . Wed. 8:00 am

Nonlinear Optics and Beam Combining V

Chair: **Robert A. Fisher,** R.A. Fisher Consulting

Invited Paper: **Phase pulling in transient Raman amplifiers,** M. D. Duncan, R. Mahon, L. L. Tankersley, J. F. Reintjes, Naval Research Lab. . . . . [874-26]

Four-wave mixing in cesium vapor, R. St. Pierre, A. Horwitz, J. Brock, TRW, Inc. . . . . [874-27]

Invited Paper: **Adaptive optic phase compensation of an aperture combined Raman laser,** J. R. Oldenettel, L. Cuellar, C. N. Howten, E. Newman, K. Roff, K. Y. Tang, Western Research Corp. . . . . [874-28]

Conditions for spontaneous generation of solitons in stimulated Raman scattering, C. M. Bowden, U.S. Army Missile Command, J. C. Englund, Southern Methodist Univ. . . . . [874-29]

***Coffee Break . . . . 10:10 to 10:30 am**

New applications and designs for deformable mirrors, E. S. Bliss, J. R. Smith, R. L. Miller, Lawrence Livermore National Lab . . . . [874-30]

Atmospheric effects on target detection with an imaging radiometer, T.-S. Chu, AT&T Bell Labs. . . . . [874-31]

Search techniques for wavefront estimation by phase retrieval, M. E. Dorros, AT&T Technologies; R. A. Gonsalves, Tufts Univ. . . . . [874-49]

**Coffee will be served in the Hilton Pavilion and in the Marriott Ballroom Lobby*

Conference on Lasers and Electro-optics '88 (Anaheim, CA, April 24-29, 1988) papers WM22 and WM23.

WM13 Suppression of Feedback-Induced Noise in Semiconductor Lasers by a Combination of Optoelectronic Negative Feedback and High-Frequency Superimposition. Noriyuki Yoshikawa, Mitsuo Tamura, Ken Hamada, Masahiro Kume, Hirokazu Shimizu, Gota Kano, Iwao Teramoto, *Matsushita Electronics Corporation, Japan.* A high reduction of the optical feedback-induced intensity noise of semiconductor lasers has been successfully achieved by a combination of optoelectronic negative feedback and high-frequency superimposition, this being useful for optical disk systems.

WM14 Low-Frequency Fluctuations and Chaos in a Distributed Feedback Semiconductor Laser with Optical Feedback. J. Mork, *Technical U. Denmark;* K. Kikuchi, *U. Tokyo, Japan.* An experimental investigation of the route to chaos in a distributed feedback semiconductor laser with optical feedback is reported. A chaotic state may be reached through intermittent switching between high- and low-frequency fluctuations.

Nonlinear Optics, Phase Conjugation, and Spectroscopy

WM15 Measurement of Raman Gain Coefficients of Hydrogen, Deuterium, and Methane. John J. Ottusch, David A. Rockwell, *Hughes Research Laboratories.* Using a single Nd:YAG laser to pump a Raman oscillator and amplifier, we measured the steady-state gain coefficients of H_2, D_2 and CH_4, at 532 nm. The oscillator spectrum and the effects of oscillator-amplifier pressure mismatch were also investigated.

WM16 Phase Conjugation in Liquid CS_2 Using a CO_2 Laser. P. E. Dyer, J. S. Leggatt, *U. Hull, U.K.* Degenerate four-wave mixing in liquid CS_2 using a TEA CO_2 laser has resulted in a phase-conjugate reflectivity of 1[illegible]. Dramatic pulse reshaping and lengthening is observed and a detailed mathematical model proposed.

WM17 Nonlinear Optical Ranging Imager. Ian McMichael, Monte Khoshnevisan, Paul H. Beckwith, *Rockwell International Science Center.* A new method that can be used to image 3-D objects in two dimensions using nonlinear optical two-wave mixing techniques is described and demonstrated. Information about the third dimension of depth is represented as an intensity modulation in the image.

WM18 Laser Beam Combining Using Near-Resonance Nonlinear Dispersion. C. A. Glosson, C. D. Cantrell, *U. Texas at Dallas;* Jay S. Chivian, W. D. Cotten, *LTV Missiles & Electronics Group.* Near-resonance nonlinear dispersion is used to create a periodically modulated index of refraction in a collection of three-level systems, acting as a grating for beam addition.

WM19 Coherent Beam Coupling and Pulsations in Self-Pumped $BaTiO_3$. Putcha Venkateswarlu, P. Chandra Sekhar, H. Jagannath, M. C. George, M. Moghbel, *Alabama A&M U.* Beam couplings and coherent pulsations in $BaTiO_3$ using two coherent beams from Ar and He-Ne lasers are studied in three configurations. Relative strengths of self-pumped and cross-coupled Bragg reflected beams are obtained.

WM20 Four-Wave Mixing at Optical Frequencies in Collisional Plasmas. L. Zhang, *UC-Davis;* E. J. Beiting, *Aerospace Corporation.* Four-wave mixing in a collisional plasma was studied theoretically for plane waves input at two frequencies. Expressions for the intensities at the six sum and difference frequencies were obtained in terms of the electron density and plasma temperature.

WM21 *Ab initio* Theory of Stimulated Rotational Raman Scattering for Diatomic Molecules and Numerical Simulation. C. G. Parazzoli, D. M. Capps, *Hughes Aircraft Company.* The time-dependent semiclassical theory of stimulated rotational Raman scattering is presented. It includes multirotational lines, Stokes, anti-Stokes, multiphoton processes, pump-population depletion, spontaneous emission. Results from a numerical code with diffraction are reported.

WM22 Numerical Studies of Transient Raman Amplification. Godehard Hilfer, *SAIC;* Curtis R. Menyuk, *U. Maryland;* John Reintjes, *U.S. Naval Research Laboratory.* The (2 + 1)-dimensional Raman amplifier code RAM2D1 was used to study numerically the effects of the Raman interaction observed in the Naval Research Laboratory experiments. Both transient and diffractive effects are included in the code.

WM23 Transient Pulse Evolution in the Long Distrance Limit of Stimulated Raman Scattering. Curtis R. Menyuk, *U. Maryland;* Godehard Hilfer, *SAIC.* We consider the evolution of transient SRS pulses over lengths sufficiently long to substantially deplete the pump. We show that both the pump and the material excitation develop rapid oscillations which can be described analytically.

WM24 Establishment of Phase in Stimulated Brillouin Scattering Beam Combiners. Joel Falk, Morton Kanefsky, Ronald Mehringer, Paul Suni, *U. Pittsburgh.* The phase difference between two stimulated Brillouin scattered beams generated in a single material must be described as a random process whose probability distribution depends on the overlap between the two pump beams, the Stokes pulse width, and the phonon lifetime.

WM25 Experimental Studies on the Second Harmonic Generation of Broadband High-Peak-Power Laser Radiation at 527 nm Using a Quadrature Crystal Array. M. S. Pronko, S. P. Obenschain, R. H. Lehmberg, *U.S. Naval Research Laboratory.* We present experimental results on the production of broadband laser radiation at a wavelength of 527 nm. A two-crystal quadrature configuration is shown to have higher second harmonic conversion efficiency than conventional single crystal systems.

"Formation of soliton-like structures in stimulated Raman scattering," (C.R. Menyuk), submitted to the International Quantum Electronics Conference '88 (Tokyo, Japan, July 18-22, 1988).

Formation of Soliton-like Structures in Stimulated Raman Scattering

Curtis R. Menyuk
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228
tel. no. (301)-455-3488

ABSTRACT: Conditions for the formation of soliton-like pulses in stimulated Raman scattering are derived. Use of the spectral transform method to study arbitrary initial pump shapes is described with a concrete example.

Formation of Soliton-like Structures in Stimulated Raman Scattering

Curtis R. Menyuk
Department of Electrical Engineering
University of Maryland
Baltimore, MD 21228
tel. no. (301)-455-3488

The equations which describe transient stimulated Raman scattering are[1]

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -i\frac{k_L}{k_S}\kappa_2 Q E_S,\\ \frac{\partial E_S}{\partial z} &= -i\kappa_2 Q^* E_L,\\ \frac{\partial Q}{\partial t} + \Gamma Q &= -i\kappa_1 E_S^* E_L.\end{aligned} \tag{1}$$

Here, E_L and E_S are the complex amplitude envelopes of the pump and Stokes waves, k_L and k_S are the corresponding wavenumbers, Q is the material excitation, κ_1 and κ_2 are gain coefficients, and $\Gamma = T_2^{-1}$ is the material damping rate. While these equations have been extensively examined in the past, their evolution depends strongly on the initial conditions, and there remains much that is of substantial interest to be examined.

If the initial pump pulse is larger in width than T_2, it is possible for solitons to form when the phase of the Stokes pulse undergoes a rapid phase flip.[2-4] How rapid must this flip be? Here, we address this question.

We first note that if $T_w \gg T_2$, where T_w is the width of initial pump wave, then Eq. (1) reduces to

$$\begin{aligned}\frac{\partial E_L}{\partial z} &= -\frac{k_L}{k_S}\frac{g}{2}|E_S|^2 E_L,\\ \frac{\partial E_S}{\partial z} &= \frac{g}{2}|E_L|^2 E_S,\end{aligned} \tag{2}$$

where $g = 2\kappa_1\kappa_2$. Equation (2) is easily solved. Letting $A_L = |E_L|$, $A_S = (k_L/k_S)^{1/2}|E_S|$, and noting that

$$K(t) = |E_L(z,t)|^2 + \frac{k_L}{k_S}|E_S(z,t)|^2 \tag{3}$$

is constant in z, we find

$$\ln\frac{K^2 - A_S^2}{A_S^2} = \ln\frac{A_L^2}{K^2 - A_L^2} = C - 2gK^2 z \tag{4}$$

at each point in time, where C is a constant of integration. We now suppose that E_S has the form

$$E_S = K\Gamma_S t + iK_S \tag{5}$$

in the neighborhood of $t = 0$, where $K(t) = K$ is constant. We assume that $K_S \ll K$ but is non-zero, resulting in a small deviation from an exact π-phase shift for E_S. In the neighborhood of $t = 0$, we then find

$$C = \ln\left[\frac{K^2(1 - \Gamma_S^2 t^2) - K_S^2}{K^2\Gamma_S^2 t^2 + K_S^2}\right] \simeq \ln\left[\frac{K^2}{K^2\Gamma_S^2 t^2 + K_S^2}\right], \tag{6}$$

so that

$$\frac{A_L^2}{K^2 - A_L^2}\frac{K^2\Gamma_S^2 t^2 + K_S^2}{K^2} = \exp(-2gK^2 z) \equiv F(z). \tag{7}$$

We conclude

$$[A_L(t = 0)]^2 = \frac{FK^2}{F + K_S^2/K^2}. \tag{8}$$

Defining τ as the t-value at which $[A_L(t)]^2 = \frac{1}{2}[A_L(0)]^2$, we find that

$$\tau = \frac{1}{\Gamma_S}\left[\frac{1}{2}\left(\frac{K_S^2}{K^2} + F\right)\right]^{1/2}. \tag{9}$$

For a soliton-like structure to form, it must be the case that $\tau < T_2$ at some z-value. Noting that $F \to 0$ as $z \to \infty$, we conclude that this will occur if

$$\frac{\Gamma}{\Gamma_S}\frac{K_S}{K} < 1. \tag{10}$$

In experiments where a Pockels cell was used to impose a phase reversal, soliton-like structures were only observed intermittently.[4] Equation (10) and a careful reading of the experimental papers suggests that the ratio Γ/Γ_S may have been too small to lead to a reasonable expectation of satisfying Eq. (10).

We now turn to consideration of the case where the pulse size is small compared to T_2. This limit is of substantial interest because in recent experiments at the Naval Research Laboratory, pulse sizes of about 40 picoseconds and T_2-values of about 600 picoseconds are typical.[5,6] In this limit, Γ may be set equal to 0 in Eq. (1). The equations are then integrable using spectral transform methods.[7,8] However, the usual method of solution must be substantially modified, leading to substantial modifications in the behavior of the solutions.

In carrying out this theory, it is useful to first normalize our variables so that

$$\begin{aligned} \frac{\partial A_1}{\partial \chi} &= -XA_2, \\ \frac{\partial A_2}{\partial \chi} &= X^* A_1, \\ \frac{\partial X}{\partial \tau} &= A_1 A_2^*, \end{aligned} \tag{11}$$

where A_1 and A_2 are the normalized pump and Stokes amplitudes, X is the normalized material excitation, and χ and τ are normalized distance and time. We now define two new quantities, u_1 and u_2 which satisfy the equations

$$\begin{aligned} \frac{\partial u_1}{\partial \chi} - i\lambda u_1 &= Xu_2, \\ \frac{\partial u_2}{\partial \chi} + i\lambda u_2 &= -X^* u_1, \end{aligned} \tag{12}$$

and

$$\begin{aligned} \frac{\partial u_1}{\partial \tau} &= -\frac{i}{\lambda} S_3 u_1 + \frac{1}{\lambda} S_+ u_2, \\ \frac{\partial u_2}{\partial \tau} &= \frac{i}{\lambda} S_3 u_2 - \frac{1}{\lambda} S_- u_1, \end{aligned} \tag{13}$$

where

$$\begin{aligned} S_3 &= \frac{1}{4}(A_1^* A_1 - A_2^* A_2), \\ S_+ &= \frac{i}{2} A_2^* A_1, \\ S_- &= S_+^*. \end{aligned} \tag{14}$$

Equations (12) and (13) are only *compatible*, *i.e.*, their cross-derivatives are equal, only if Eq. (11) holds.

In the usual spectral transform approach, as applied for instance to the nonlinear Shrödinger equation, one would proceed by defining scattering data which relate (u_1, u_2) at $\tau = +\infty$ to (u_1, u_2) at $\tau = -\infty$. This scattering data has a one-to-one correspondence with the original variable set. Moreover, since u_1 and u_2 evolve simply in χ at $\tau = \pm\infty$, so do the scattering data, and one can then infer the evolution of the original variable set.[9] In our case, a fundamental difficulty results from the fact that X does not in general tend toward zero as τ tends toward $+\infty$, and the χ-evolution at $+\infty$ is not simple. Kaup[7] has resolved this issue in certain important cases by showing that our variable set only depends on the evolution of u_1 and u_2 at $\tau = -\infty$.

We apply his approach in detail to cases of practical interest to determine the full nonlinear evolution, notably the case where the initial pump and the initial Stokes have the same shape. We also show that in contrast to the usual case, where the initial pulse decomposes into a set of enduring solitons and a dispersive continuum, all soliton-like structures must be transient. From a physical standpoint, this result is almost self-evident. These soliton-like structures are well-known to possess a velocity slower than light.[2] They must therefore ultimately disappear at the back end of the pulse.

This work was supported by the Naval Research Laboratory, and the author is grateful to Dr. J. Reintjes for numerous, useful discussions.

REFERENCES

1. R. L. Carmen, F. Shimuzu, C. S. Wang, and N. Bloembergen, Phys. Rev. A **2**, 60 (1970).
2. K. Druhl, R. G. Wenzel, and J. L. Carlsten, Phys. Rev. Lett. **51**, 1171 (1983).
3. K. J. Druhl, J. L. Carlsten, and R. G. Wenzel, J. Stat. Phys. **39**, 615 (1985).
4. R. G. Wenzel, J. L. Carlsten, and K. J. Druhl, J. Stat. Phys. **39**, 621 (1985).
5. M. D. Duncan, R. Mahon, L. L. Tankersley, and J. Reintjes, "A Study of Transient Stimulated Raman Scattering in Hydrogen," (preprint).
6. J. Reintjes, M. D. Duncan, G. Calame, L. L. Tankersley, and R. Mahon, Optics News **39**, 101 (1987).
7. D. Kaup, Physica **6D**, 143 (1983).
8. H. Steudel, Physica **6D**, 155 (1983).
9. See, *e.g.*, M. J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering Transform*, (SIAM, Philadelphia, 1981).

END
DATE
9-88
DTIC